DIPLOMA PORJECT (2003-2004)

Far-field Terrain Navigation

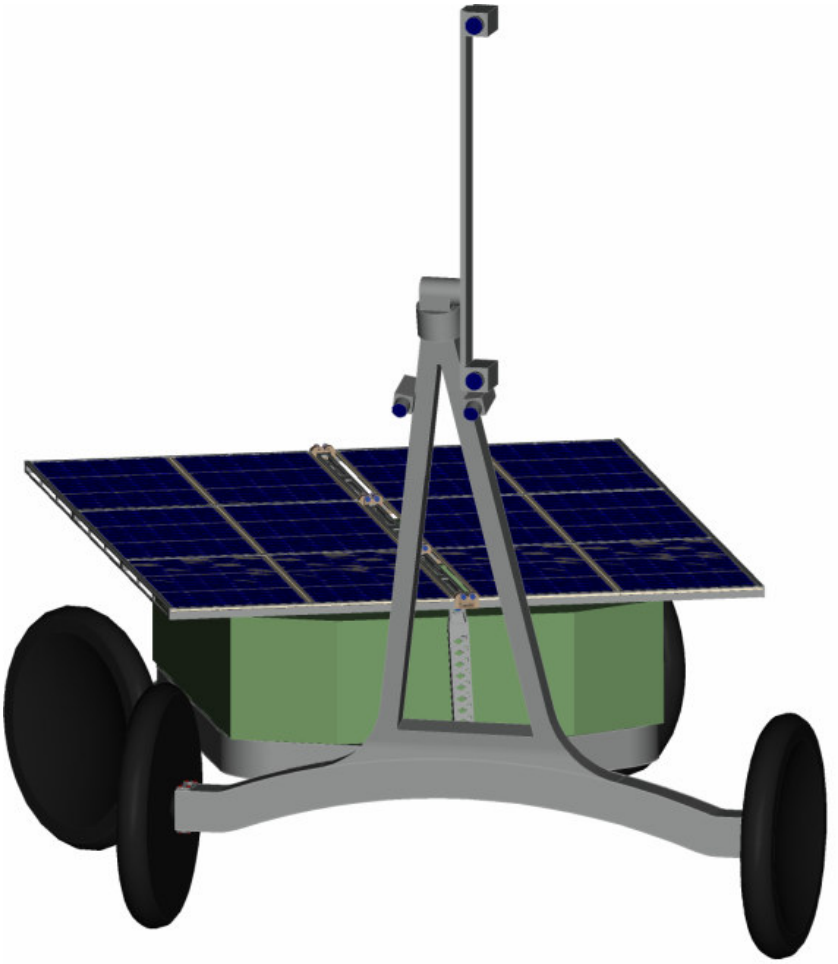
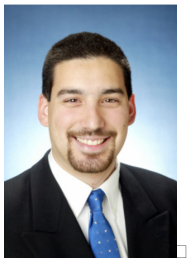


Figure 1 Futur rover, with vertical wide basline, called Zoë



PROFESSEUR : PROFESSEUR REYMOND CLAVEL
ASSISTANTS : DR D. WETTERGREEN, DR C. BAUR, DR T. FONG, M S. GRANGE
STUDENT : ALEC AVEDISYAN

Table of contents

1 Introduction

The Life in the Atacama project began two years ago. The main idea in this project is to find (trace de vie) life in an undiscovered area of space. The rover has to go from a defined start region to a destination region, by avoiding all existing obstacles by itself.

1.1 Project Objectives

The aim of the project is to be able to navigate in an unknown environment in order to simulate at best the conditions and the constraints that one could meet on the planet Mars. That is why the site of the Atacama Desert has been chosen. There are areas in the interior of the desert where no one has detected life., and the terrain is similar to the one we could find on Mars.

Concerning the navigation on the rover, several levels of sophistication have already been overtaken, as we are going to see it further. The current navigation system uses near field(0-7m) and satellite imagery (30-infinite) as we can see on the Figure 2.

Our Navigation system has to fill the gap between two areas information ■ :
Near field (0-7m) ■
Satellite imagery (30-infinite) ■

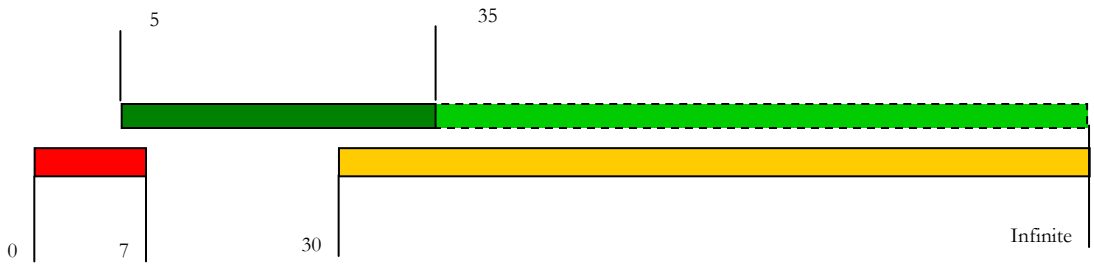


Figure 2 Illustration of the gap in information

The main goal for my part in this project is to make a navigation system, based on the rover's sensors (Camera (single) (or/and) Camera (stereo) (or/and) Laser), to identify obstacles and target which are in the middle range.

The output that we will get from this analysis is a value of the terrain traversability and also a classification of detected terrain textures (physical properties).

To reach our goal, two ways were proposed:

1.1.1 Semantic and Topologic (Toposemantic)

By "toposemantic", we mean detection of topological relationships between objects (terrain patches, obstacles) according to their semantic definition.

The implementation of the toposemantic interpretation can result in a search for color, search for continuity, regional similarity, texture in the picture (photo, or laser scan), and any other treatment on optical flow and images processing.

1.1.2 Geometric

The geometrical method is more quantitative; it models Cartesian space geometrically, from the data received by different sensors. We can use different sources of information such as stereo pairs or laser to generate precise geometric models. It could even be possible to mix either information's (laser & (mono or stereo)) to build volumes. The odometry brings also lot of information.

1.1.3 Advantages and disadvantages of each method

The advantage of the semantic method is that it requests much less time of computing. It is more independent in time, which means that the future information doesn't need to be dependent on the past or present information. This contrarily to the geometric method which builds the ground as it is discovered and which must know an evolution on the scene.

The additional advantage of the geometrical methods is precisely a modeling of the ground which can be translated by a map much more precisely than the one given by a digital elevation model (satellite). The utility of this kind of map can be large if we plan to make other moves on the same ground in a short- or medium-term future, because the modifications of landscape are frequent in the long term in a desert (sand dune) or on the planet Mars (meteorite).

At this point the real question that we have to ask is:

What are really the needs for the rover's navigation?

- *Obstacle avoidance, detection of change of ground according to the forecast (satellite imagery)* [more semantic]

- *Or, a meticulous cartography of the ground (for some future crossings)* [more geometric]

We can also make cartography with the semantic approach and vice versa, but this is more difficult.

1.1.4 30 meters or infinite?

Here also we have to make a choice, the vision Far Field will enable us to have a vision until infinite. But it is necessary to treat the additional information which will cover information already existing and given by satellite imagery? Or is it better to stop just at the level to close the existing gap?

I will be personally more interested to use all information until infinite. The use of the skyline can be a very good indicator (information).

1.2 Existing work in the project

Here, we have an overview of the existing project with all the modules that take part to it, we are not going to describe here each one of them, but point out the ones, which have a direct interaction with our system

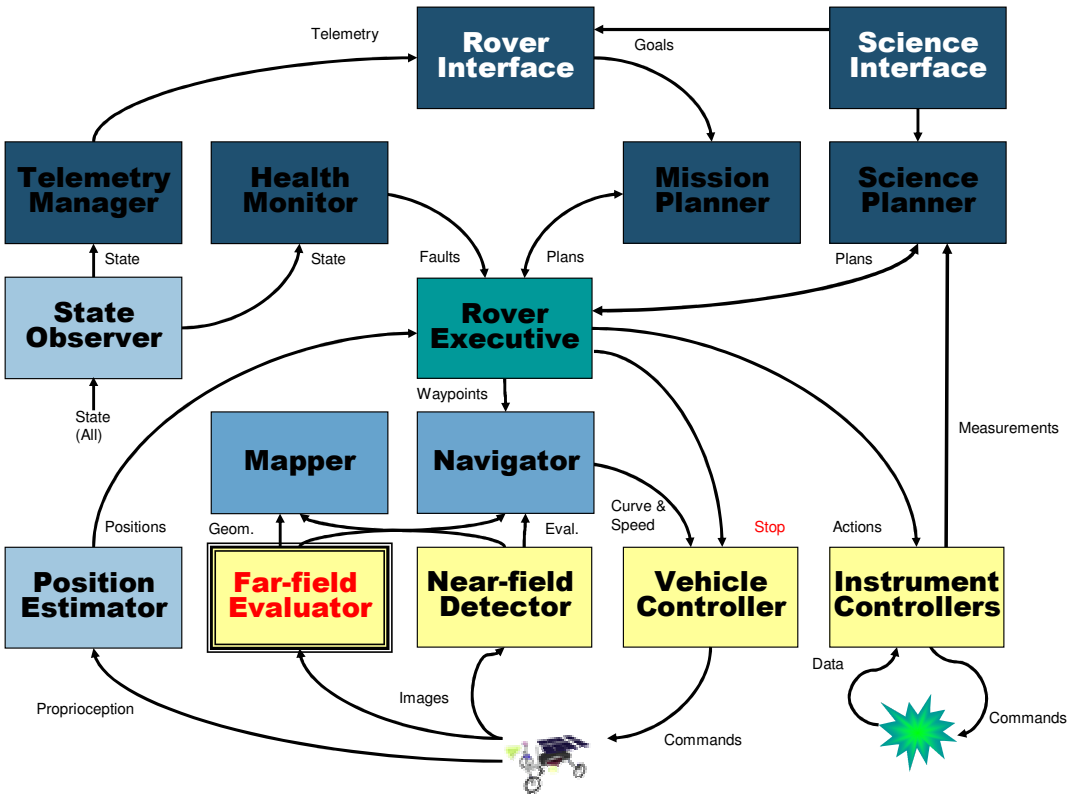


Figure 3 Software Organization Schema

We notice that we face a singularly complex system with distinctive modules. The module that we are going to use in our project is the **Far-Field Evaluator**, which will be like the Near-field Detector indirect interaction with the Mapper and the Navigator. The goal of the Mapper is to aggregate and register environmental sensing to build maps. This will include geometric information, visual information (color, texture) and derived information (geology). The navigator generates motion commands to avoid obstacles and reach goals. Navigators operate on composite ground evaluation and utilize onboard sensing to avoid near-field obstacles and allow continuous motion. It also selects arcs based on speed, obstacle height and goal location [RSEK94]. (See State of the art: **Navigation**)

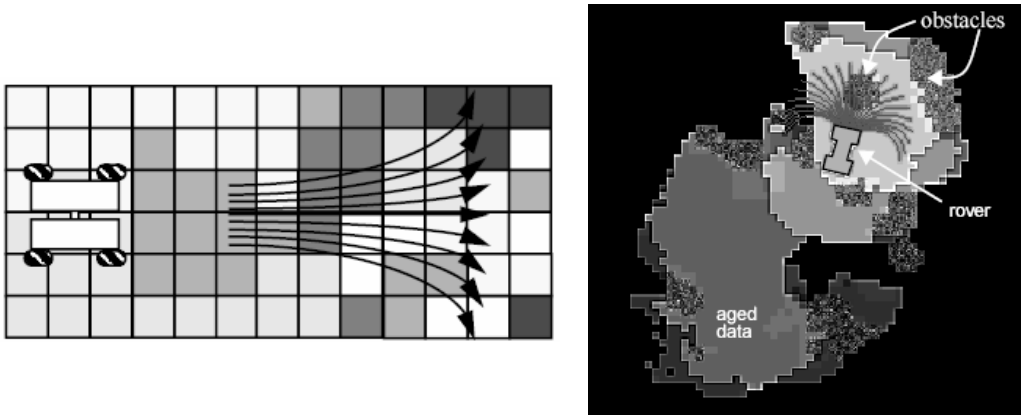


Figure 4 Navigator - arcs processing

1.3 Constraints

The rover has physical constraints that one will have to respect. The most important one is that it has to be completely autonomous. This implies energy reserves and therefore a minimization of the use of energy. In order to guarantee this constraint, we need elements on the rover, which does not absorb a lot of energy and very light too, with a minimum of redundancy. (i.e. for the sensors) .

We are, for example, going to prefer having a unique camera, on which we can make specific treatment of picture instead of having several specialized cameras which would filter a unique domain of frequency.

The sensors at our disposal are a pair of camera SONY dfw-sx900, which are high resolution and very good quality. The laser is a SICK LM20 of high accuracy functioning in 1D.

Figure 5 Available sensors on the rover for far field navigation.

Finally, in our case we have another constraint, which is the processing time of the picture. We do not need real time because we count on the fact that the rover takes a picture every 10[m]. We have estimated the constraint of time processing picture at 2-3 [s]. (Rover speed: 10[m/s], CPU utilization; less than 30%)

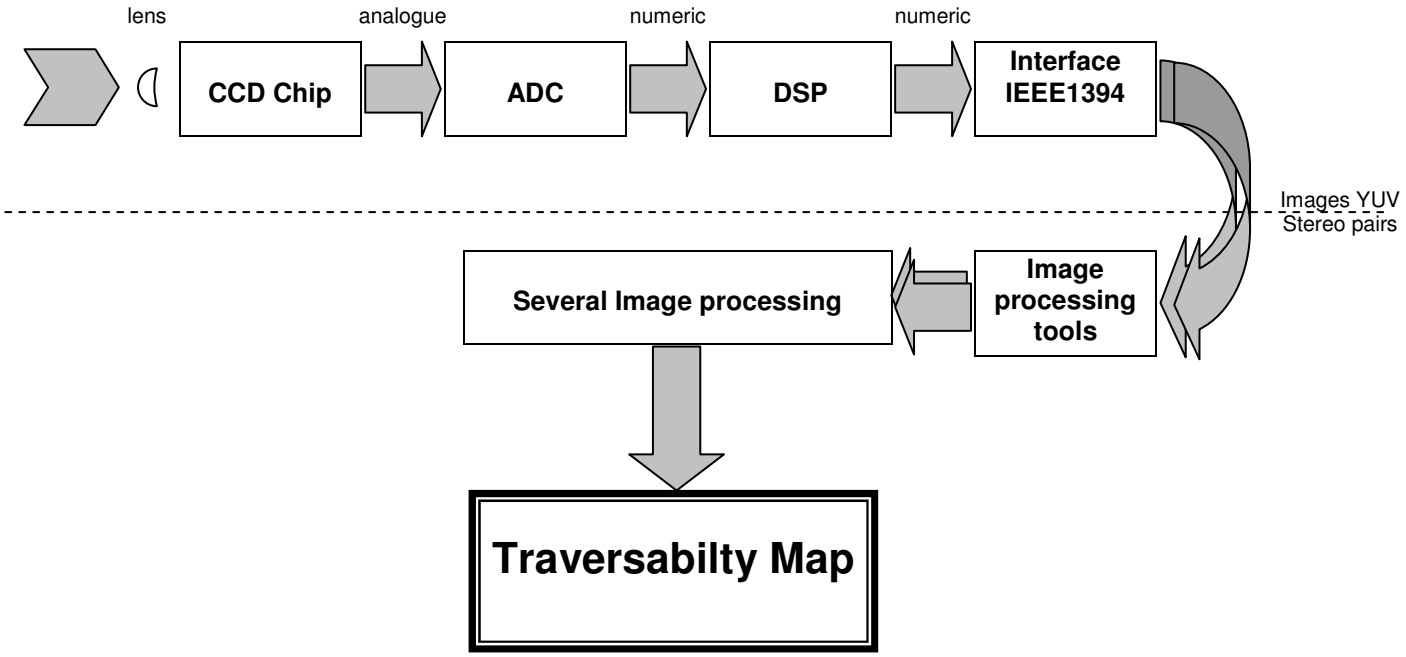


Figure 6 General picture acquisition and processing mechanism

1.4 Summary

In this project, we propose a new way to carry out robot navigation considering long-range visual information. The goal is to generate a traversability map by identifying dangerous and/or interesting areas in distant terrain. To achieve this, we rely on imagery taken by the rover and various processing techniques on these data.

We first apply novel geometric techniques to the stereo pairs to segment the images based on distance. This segmentation is an indispensable preprocessing step that the following phases will rely upon. Using the horizon as a reference, we generate a coarse depth map of the scene. The depth map is then used to identify regions in the image such that all points within a region are at a comparable distance.

Next we apply several toposemantic techniques to the segmented bands within one of the images. Using statistical analysis, morphological operators, and a variety of filters, we generate three principal metrics. These estimates of roughness, textures and occlusion, combined in a traversability map, will be further refined to generate a single representation of the visible terrain.

Our final goal, the traversability map, is the result of toposemantic and geometric processing techniques. Using these new techniques, far-field navigation augments existing near-field obstacle avoidance in the navigation system. It accepts high-resolution long-range stereo imagery as input and produces this comprehensive traversability map.

2 State of the art

In this section, we will look at most of the existing references with like principal topic, the existing navigation system on the "Life in the Atacama project", monocular image processing, stereo vision and distance detecting sensors. The implementation methods and algorithm will not be describe in details.

We will divide the state of the art in four sections. We will start by topological and semantic methods that we will call toposemantic. Then we will go on with the geometrical method. We will also have a look on the methods that use the two above together. And we will finally show what exists in the project for navigation. One can also notice that there have been very few works carried out about far field

2.1 Toposemantic

Image processing in a semantic [BSM01][DFJP02][RHOM95][W01][W02][JMSB01] way is so vast that it would be useless to make an exhaustive list of all the possibilities. We can make a general regrouping of the different existing methods that we are then going to use:

- Statistical analysis
    - Intensity Histogram
    - Classification
    - Connected Components Labeling
- Morphological operators
    - Dilation, Erosion, Opening
- Filters
    - Mean Filter, Median Filter
    - Gaussian Smoothing, Conservative Smoothing
    - Frequency Filters
    - Laplacian/Laplacian of Gaussian Filter
    - Unsharp Filter

Each of the methods expressed in the above list has been implemented and used in our project.

Regarding the method that we will call topological treatment, there are a few number of methods. We selected the ones we consider the most interesting and which are used generally for shape or texture.

We selected the following operators:

- Image Transforms
    - Hough Transform
    - Fourier Transform
    - Gabor filter

We had to make the selection of the operator, which is the best for our project. We knew by advance that the Hough transform would be incompatible with the type of picture that we have because this one is the most commonly used for the detection of regular curves such as lines, circles, ellipses.

The Fourier Transform is an important image-processing tool, which is used to decompose an image into its sine and cosine components. The output of the transformation represents the image in the Fourier or frequency domain, while the input image is the spatial domain equivalent. In the Fourier domain image, each point represents a particular frequency contained in the spatial domain image. A method exists, which optimizes the time of computation and reduces the complexity to $N \log_2 N$ instead of N^2, This method is called the Fast Fourier Transform (i.e. FFT and FFT2 for two dimension).

The last method selected is the Gabor filter [KKKM97] [RKO95]. A Gabor filter is a sinusoidal function to which we have added a Gaussian envelop. On the frequential plan, this function transforms itself into Gaussian. The sinusoidal function is characterized by its frequency and by its orientation. Thus, the Gabor filter can be seen as edges detector of particular orientation, because it will react to perpendicular edges in the direction of sinus propagation. The frequency of the sinus, indicates to which frequencies the filter will be sensitive and react. This method does not suit neither exactly to the needs that we will need later; and for three reasons : The first one is that the information of the orientation is useless for us, the second reason is that the processing time is much more important than the two previous methods because one has to make a convolution for each of the samples from a existing sample database, and the last reason is that we need a database in advance.

That is why we chose without any hesitation to work with FFT.

2.2 Geometric

2.2.1 Hybrid systems (Stereo and laser)

Figure 7 Nomad Rover

The needs in autonomous navigation are generally limited to close distances. For example, the robot navigation, which moves in closed areas or further down in the case of NAVLAB (which can even extract distances at more than 40 [m]). The extracting methods of information in order to make far field – distant field navigation (according to Figure 12) are rare. This can also be explained by the fact the exploration of big spaces is not usual. If we take for example the "Nomad case", the meteorites searcher robot in Antarctic (Figure 7), it uses two pairs of two wide angle CCD cameras for navigation and also a laser rangefinder mounted on the stereo camera mast. These sensors are used by the navigation system to detect obstacles on the path of the robot. It scans only several meters in front of the robot to find features that the stereo cameras could miss.

2.2.2 Laser

In the case of NAVLAB II [SSBD99], the laser is used for navigation. The technique consists in making an horizontal scanning, which will give us 2 distant lines in order to recuperate the lengths and then report the results on a graph (represented on Figure 8 b) (scan at 180°).

Unfortunately, the use of laser gives us no information relative to the texture and roughness of the ground. It just tells us if there is an obstacle or a "local" discontinuity.

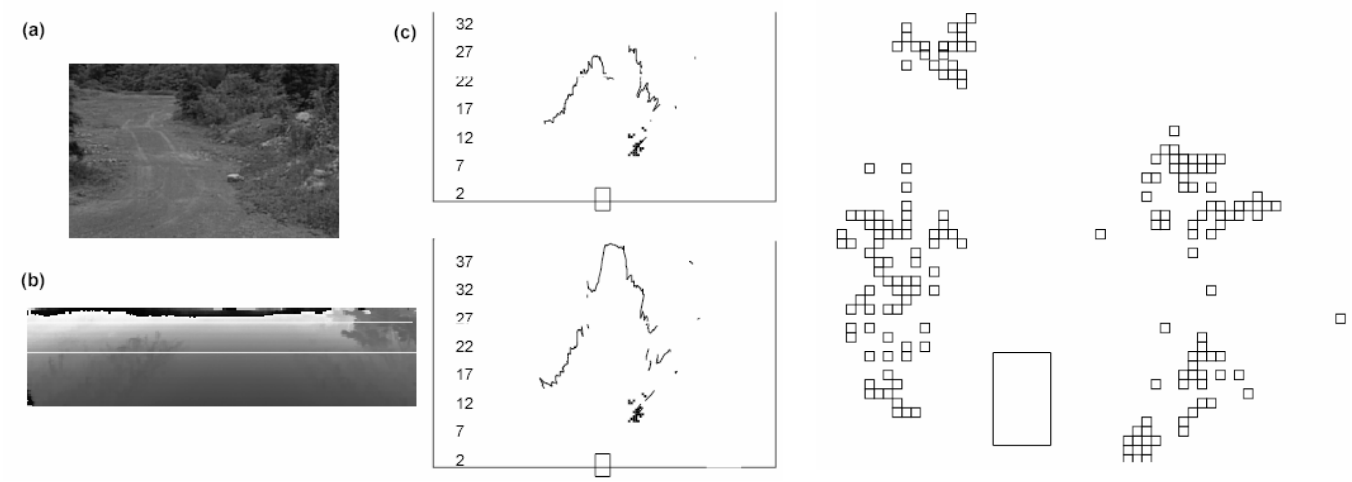


Figure 8 Left : a) Original picture, b)Scanline-Based Processing from a Range Image c) Detected distances for both picture, Right A Local Map [SSBD99]

2.2.3 Stereo

Stereo is a vast domain in which many things have been discovered at this day. Regarding our case, we are going to study more particularly the wide baseline stereo. Researches have been recently carried out in the wide-baseline field for the navigation on Mars [CFO03]. They implemented a very interesting method, which consists in calculating the displacement distance of the robot with an odometer. It uses this information as a baseline for both picture taken with a monocular camera. But this introduces another type of problem: First, stereo algorithms typically calibrate a pair of stereo cameras such that the relative position and orientations of the cameras are known to high precision. This is not possible with this kind of wide-baseline stereo, since rover odometry errors prevent such high precision camera positioning. Second, the change in the viewpoint for the images makes stereo matching more difficult, since the ground has no longer the same appearance on both images. To rescale the picture, they use rectification with the epipolar lines and modify the picture accordingly.

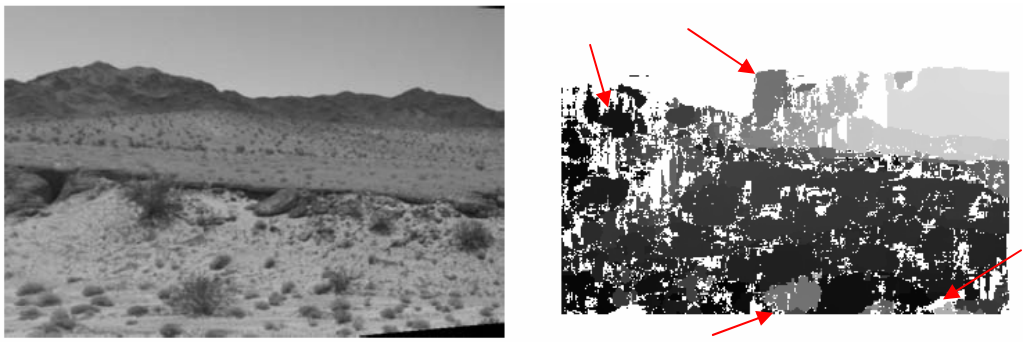


Figure 9 Left: Computed disparity map. Right: Disparity image Dark values represent larger disparities. [CFO03]

Here is the result for this method Figure 9. The white parts show us the area where there is no information relative to the distance. Moreover several areas (showed by the red arrows) give wrong information about the distance. But generally the information is of good quality.

In the same field but within the framework of navigation Near Field (less than 10[m]), Mathew Deans of NASA [MDE04] (february 2004), made research to do navigation using SIFT algorithm made research in the idea make navigation by using algorithm SIFT [DGL04]. The system that he want to set up is muti-camera (3 cameras with position and width of different focal distances). This project is one of rare to use SIFT within the framework of navigation. In our project we use SIFT within the framework of navigation Far Field, and also in the idea to make extraction of distance by using the stereo vision

2.3 Navigation

Regarding navigation, more than two modules exist, as seen above : The « Global Navigator » (Tempest) and the « Local Navigator ». Tempest deals with the navigation of the robot at a high scale. Tempest is based on a trajectory planning algorithm, which name is D* (D-star) [AST93] [AST94] [AST95] (Figure 10 Left) .

This algorithm finds the lowest-cost path through a graph, and plan optimal traverses in real-time, by incrementally repairing paths to the robot's state (Figure 4), as new information is discovered. The path is an optimal one if the sum of the costs of transition (cost of an arch), is minimum throughout the existing sequences in the graph. If one discovers one or several arches, which are incorrect, the rest of the path must be recalculated to keep the optimality. A complete path is only optimal if the sum of all the small paths, which are part of it are also optimal.

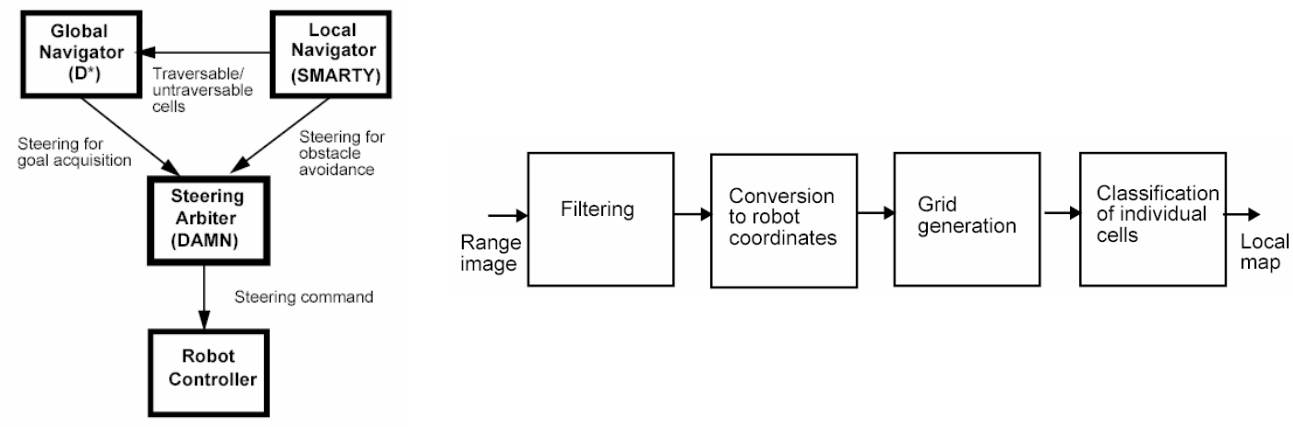


Figure 10 Left: Navigation System Diagram, Right: Range Image Processing [AST95]

The range image processing module takes a single image as input and outputs a list of regions which are un-traversable (Figure 10 Right). After image filtering, the (x,y,z) location of every pixel in the range image is computed in a coordinate system relative to the current robot position. The points are then mapped into a discrete grid on the (x,y) plane. Each cell of the grid (Figure 8 Right) contains the list of the (x,y,z) coordinates of the points which fall within the bounds of the cell in x and y. The terrain classification as traversable or un-traversable is first performed in every cell individually. The criteria used for the classification are:

- **The height variation of the terrain within the cell,**
- **The orientation of the vector normal to the patch of terrain contained in the cell**
- **The presence of a discontinuity of elevation in the cell.**

(To avoid frequent erroneous classification, the first two criteria are evaluated only if the number of points in the cell is large enough. In practice, a minimum of 5 points per cell is used.)

3 Implementations and choices for the navigation system

We enter here the heart of the subject: the research of a navigating system, which is efficient and considers the different constraints, which have been previously mentioned.

Regarding the Figure 11 some questions come to us. How to extract interesting information from a picture? How to define interesting information? How to classify and group the information? These are the questions, which have justified our choices

Figure 11 Atacam Desert

The first choice was to make a discretization of the picture. This is a cut-out of the picture relative to equidistant areas. This choice enables us to resolve a multitude of problems. For example, if we try to do a matching in a stony area, which lies at 10[m], we would not like it to be mixed up with a rocky area which would lie at 1[km], even if the two parts of the picture have the same aspect. We know that these two elements are really different and that the first one is easily practicable for the rover whereas the other one would be very dangerous.

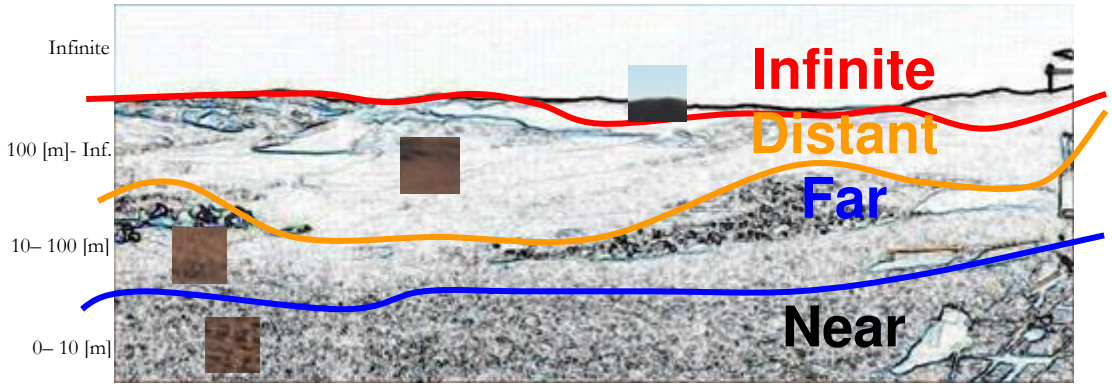


Figure 12 Equidistant picture segmentation

A first gross discretization will be as we can see on Figure 12 a subdivision of the distances in a non linear manner. The first area close to the 0 to 10 [m] range will be called « Near Field ». We are not going to treat the information, which are in this segment. It is the role of the Near Field Detector [CUR03]. The second area is comprised between 10 to100 [m] and it is in this range that we are going to concentrate our works and carry out the main analysis of our work. The following area is comprised between 100[m] and the skyline (infinite), we will call it the « Distant Field ». This part of the picture will be treated more superficially. Finally, the last area «Infinite » will be treated only in order to remove it from the picture. We will carry out a pretreatment on the picture by erasing all the information lying beyond the skyline. The sky and possible clouds can actually interfere on the treatments that will follow.

As mentioned before, the continuation of the analysis will consist in two big sections. The first part will take care of the toposemantic analysis, and the second part will be a geometric analysis. Finally, the last section, that we will call Data Fusion, will consider the two parts cited above to extract the essential information like traversability map or the Science Target map.

3.1 Part I : Toposemantic

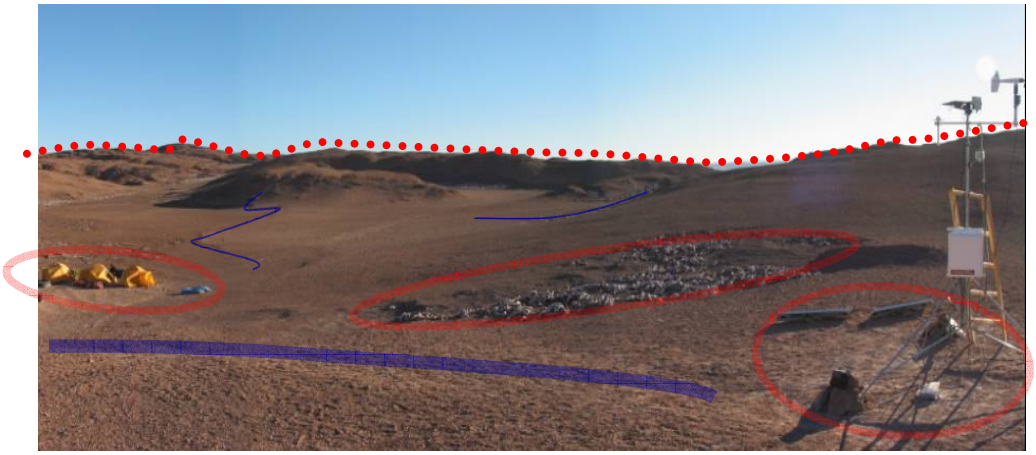
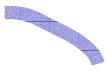


Figure 13 Toposemantic information tracking

3.1.1 Introduction

One can extract from a picture a mass of information. It is the way of treating it, which will be decisive. As one is looking closer to a picture like on Figure 13, one can notice that it would be interesting to do a multiple images processing.

The red circles show the areas being different either by texture, color or shape, from what we can see on the rest of the picture. It could be interesting to use a texture recognition algorithm, which could tell if we face a type of different ground in this case



The blue arcs, that we call occlusions, are parts of the picture, which are difficult to detect. Nevertheless, we know that after the « line», there is an abrupt change in the type of texture. It would be useless to try to detect a texture difference in the literal sense, because it would be undetectable. But we will rather think about making a density texture difference between before and after occlusion. We know for example that local maxima will have a stronger density by pixel surface before occlusion than after. The methods that we are going to apply here will rely essentially on the primary and second derives of a part of the picture (Edge detection, Gaussian difference, Laplacien, approximation of derives.).

The dotted red line shows the separation between the sky and the earth, in other words information and non-information. It will be essential to put aside this part of the picture later. The idea for this part of the picture was to find the biggest homogeneous colored areas. In order to achieve this aim, we carried more statistical analysis. We have pretty quickly forgotten the fact of working with an edge detector for several reasons that will be explained further (clouds, blurred horizon).

3.1.2 Sky Detector – Infinite

3.1.2.1 Goal

The detection of the sky is very useful, and assists us in different manners. Basically, we can use it first to detect if there is or not a sky, and then, if a sky exists we can employ it to estimate if the rover is going up or down, lean left or right, by using the value of the horizon slope.

Finally, this sky detector will be a good tool to spare image processing time, by subtracting all the Sky Area (Infinite) from original image.

3.1.2.2 Methods

Different kinds of sky detectors exist; we can for example use the color of the sky (range of blue), search for uni-colored areas, work on texture, work with low frequencies detectors etc …

As it is known, a sky can have a lot of different configurations. It can be clear or cloudy, can also have lots of different colors which do not derive from blue (sun rise/set colors), and it can also have a blurred horizon line. One can often observe these sky effects in deserts, where it is impossible for the human eye to say if this is the sky or a very far mountain Figure 14 *C*.

A *B* *C*

Figure 14 Sky Examples

Regarding these cases, my idea was to seek for the biggest areas of uniformed color. And the best way to achieve this task is to reduce the number of colors. My first tests were on 128, 64, 32, 16 colors and it was not very significant, but as we could have guessed it, the smaller the color space is, the bigger the color blobs are. Having settled these hypotheses we tried now to work on the minimal set of colors, which is 8 for RGB images (Red/No Red | Green/No Green | Blue/No Blue).

The idea is to set colors to zero or to a defined threshold.

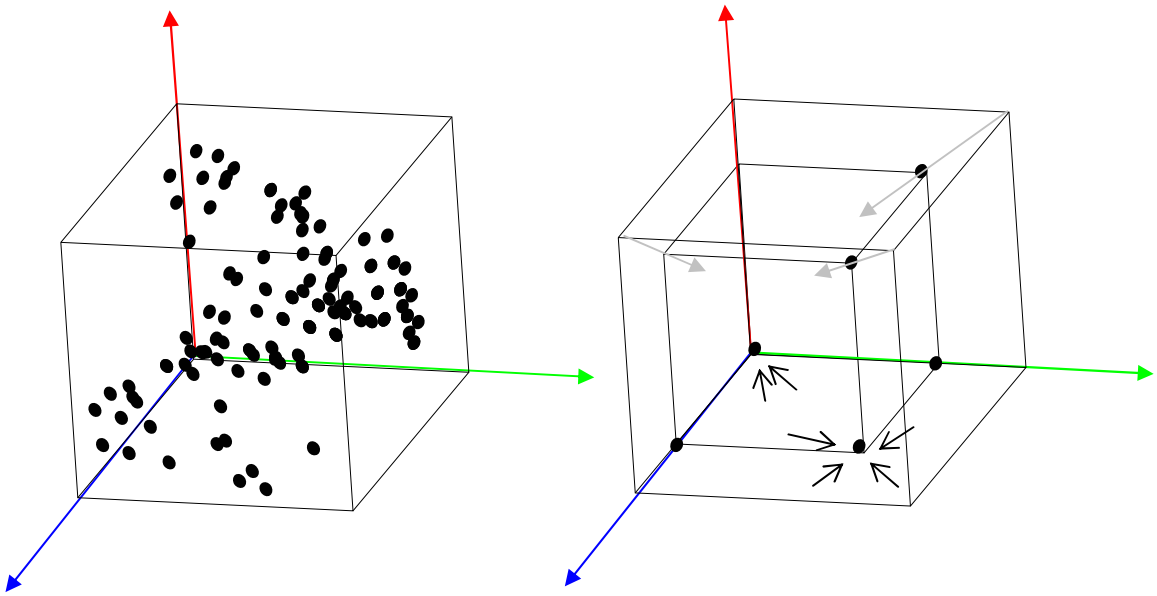


Figure 15 RGB Space of color Left picture: Dots represent each color present in the picture. Right picture show the threshold moving.

The difficulty is to find the best threshold, in order to maximize the area of color. The easiest way to do that is to calculate, in the iterative way, each possibility until we find a good result. Another method would consist in finding the eigenvector by using a Principal Components Analysis (PCA). This can be very useful if we want to optimize colors areas.

The method used here is the iterative one; it will consist in using statistical results to do a comparison between each step until a convincing result is found. We mean by convincing result a sufficient amount of sky points.

As soon as we possess the information, we can then calculate all of the interesting information like: the sky position, the slope of the sky, etc …

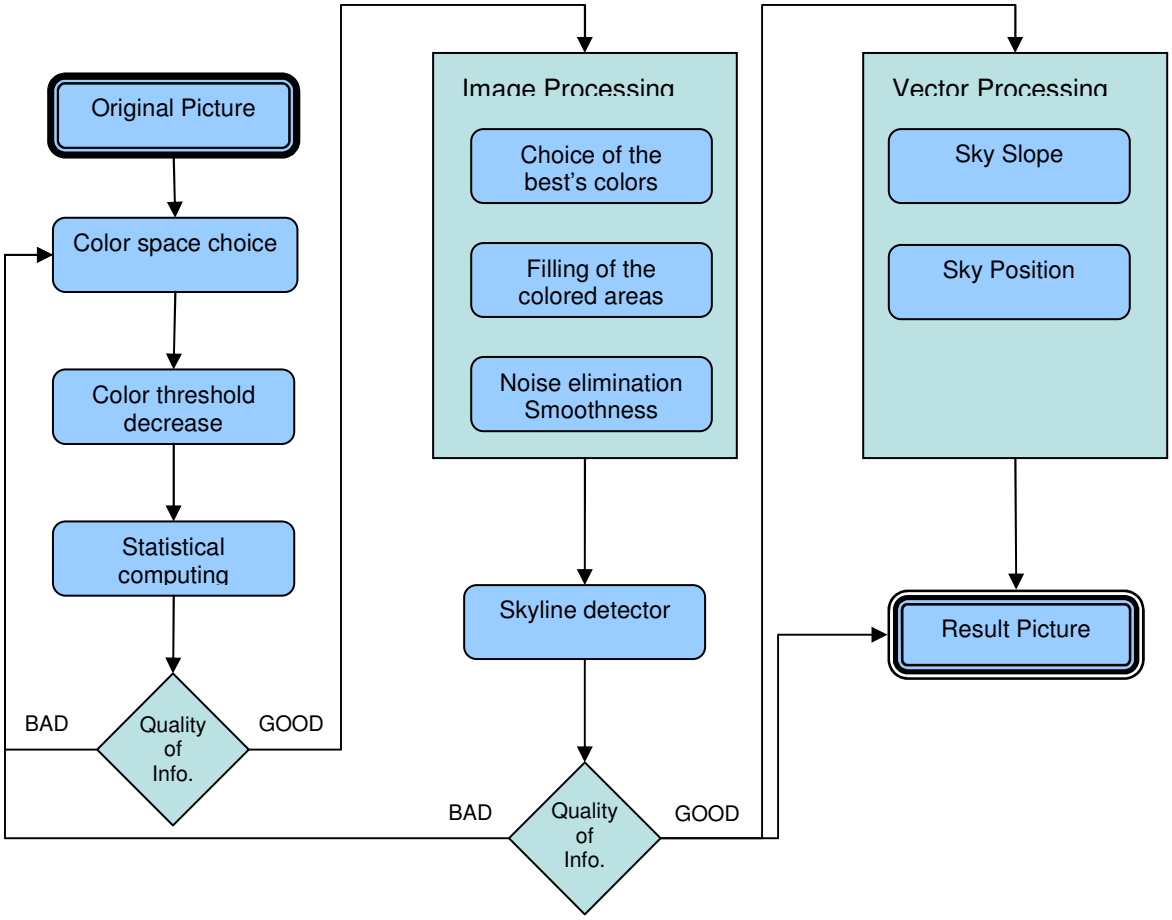


Figure 16 Diagram of the algorithm. "Quality of Info" means that there are some thresholds that decide if the information provided by several processing is enough or not.

3.1.2.3 Conclusion

Here we have a robust and simple algorithm. This algorithm is stable to color variation (it is not dependent from a defined color). It also remains relatively stable regarding the obstacles which cut the sky line and also, for regarding the skies which will not contain uniform colors such as a cloudy sky, a sunset/rise, an object in the sky.

3.1.2.4 Result

All result and comments are in Appendix A1 - Sky Detector Result

3.1.3 Terrain Classification for Traversability estimation

3.1.3.1 Goal

In this part, the aim is to do a toposemantic analysis of the picture, in order to extract the maximum of information. The main information's sought is discontinuity, either as a change of texture, ground roughness, occlusions, or color, having as a final intention to being able to categorize them as « targets » or « obstacles ».

3.1.3.2 Methods

Information classification

- How segment a picture in order to get a maximum of information about our environment?

- Does one need fixed geometrical or variable segments?

- Does one have to cover the whole picture by segmentation or just partially?

We will try to analyze these questions and demonstrate the reasons why we made the choices that will follow. It constitutes an important choice, because it will determine, in a way, the skeleton of our system. We are going to stock the information's in each segments, as in compartments and reversely we will seek other information relatively to the position of the segments. The type of information stocked will play an essential role in the decision of segmentation.

Obstacle

- Edges, Crevasses, excavations or ravines

Science Target

- Texture
- Rocks
- Ambient color
- Unusual information (like unexpected colors)

Miscellaneous

- Distances

Figure 17 Sample of landscape

In this case, we have to analyze a picture, which has a non-homogeneous and non-symmetrical information and it is difficult to discretize it. On Figure 17, there are clearly parts of the picture, which interest us, because it has a density of information (by pixel surface) much more important. We will then consider in the following examples that information's lying in the sky are null. This part of the image will automatically be rejected by using the Sky Detector.

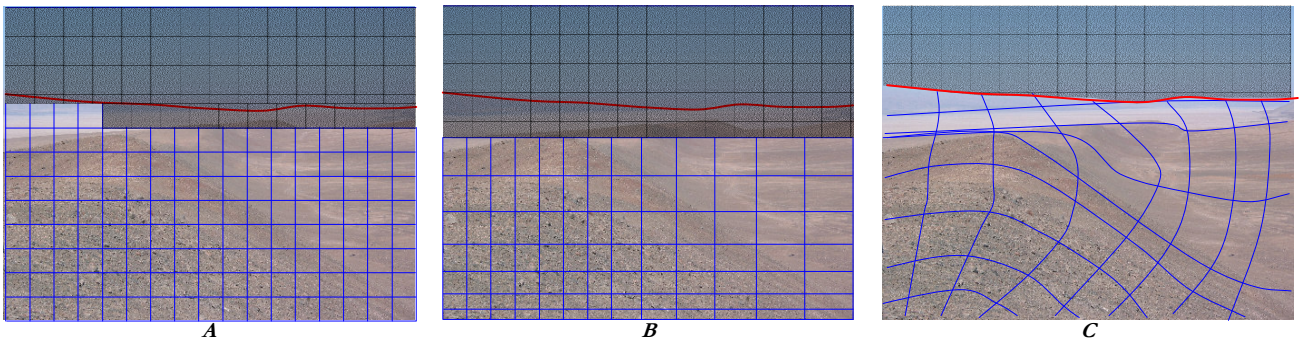


Figure 18 A: Monotonic segmentation, B: Parabolic segmentation, C: Anarchic segmentation

We have here 3 examples of segmentations, which have each their advantage, In case ***A*** each of the segments has the same size, which means that we put the same weight on a plot of land of 1[m^2] or 100 [m^2] and it is substantially difficult to interpret as long as we do not have the notion of distance. But, if we classify them with the distant information (see 3.1.10) the comparison between the segments of several pictures is efficient. The disadvantage is that a feature (like a big rock) can rely in several segments at the same time (if the segment boundary's cross the rock at the middle). Notion

In case ***B***, we placed a point, that we will call "interest center of the picture", the more a segment will be distant from the center, the bigger it will be. This method has the advantage to minimize the number of segments and concentrate on a part of the picture (optimization of calculation timing). We could also imagine a mix of ***A*** with ***B***,

which would consist in having segments like des quad tree ***B'***(identical squared basis (common to all segments) but variables sizes of segments).

Segmentation in ***C*** considers distances: we know that each of the parcels will rely in a well-known distance range (not like in ***A*** or ***B***). And we can also say that the density of information is well distributed for each segment, on the contrary it will be difficult in this case to make comparisons with other pictures because of the complexity of the stocking.

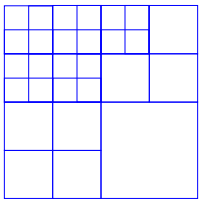


Figure 19 B': Quad tree segmentation

We will implement the project by using segmentation as in ***A*** but keeping in mind the possibility in the future to work with a type ***B'*** scale***.***

Does one have to keep all the parcels, even the ones, which are very distant? We think it is necessary to make a complete analysis about them, they can have a scientific interest (color, texture, etc..).

For each of the segments, we will keep the following information in this way:

Name	Type	Comment
Obstacle Detected	Integer	Give the size in pixel of whole the detected areas for each segment
Texture	Float vector	This is a description vector of each of the component extracted from the fast Fourier transform.
Rocks Color (HSI)	Integer vector	HSI = [0 0 0] if no rocks is detected.
Color (HSI)	Integer vector	Ambient color, major present color, HSI = [0 0 0] if no rocks is detected
Ucolor (HSI)	Integer vector	Minor present color if strongly different from **Color (HSI) ,** HSI = [0 0 0] if no rocks is detected.
Distances	Float vector	Stereo distance extraction [X Y Z]

The size of the segments is of 32x32 in our case. It is the best compromise to post information, if one consider that the dimension of the initial picture is1280x960 pixel. A Fourier transform would never give us enough information if the size was 16x16 and too many if it was de 64x64. We did the same reflection for the occlusion detector, which we will analyze further.

We will also discuss the way we will treat the datas in order to have a coherent traversability-map (see: Data fusion for traversability estimation).

3.1.4 Images neighborhood variation – Occlusion detector

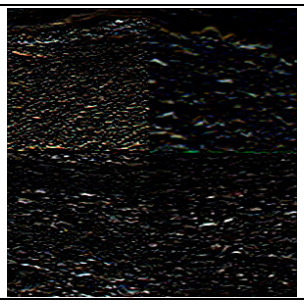
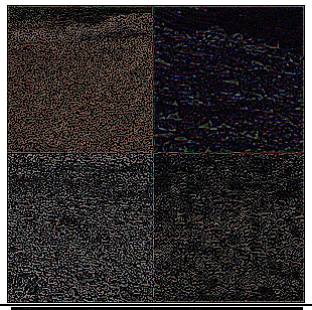
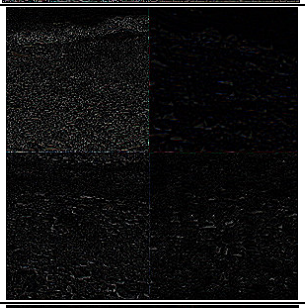
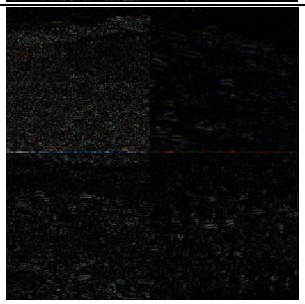
The idea is to find a method, which will enable us to find areas of the picture, which have a similarity relatively to a close neighborhood, in order to detect the discontinuities of the surface. To do so, we are going to use the fact that perspective foreshortening results in an inverse square loss of resolution with distance (i.e. far away). This method works particularly well if one finds oneself in an environment with few changing of texture as we can notice it in a desert. Therefore we know that close objects will be much more detailed than the one far away. Then, we had the idea to detect the close objects by carrying out a filter, that we will call high-pass. We will demonstrate here several possible methods and choose the best one.

Figure 20 Left: original starting samples, Right: illustration of the occlusion position

On Figure 20, we have chosen samples in order to always have a fracture on the ground, which is not easily visible to the naked eye. The pictures are taken with a fracture in same plan, which lies between 10 to 30 meters from the observer. Some fracture lines are easily seen, e.g., at regions where there abrupt changes in image color or texture (see Figure 20 both top). Other fracture lines, however, are not evident at first glance. Figure 20 both bottom, for example, shows several fractures (reported by field survey personnel), which are difficult to identify. In our case, we will treat the information in the range of approximately 0 to 100. It is useless trying to detect crevasses over this distance. We will use distances extracted from the geometrical part (see 3.2).

3.1.4.1 STEP 1 : HIGH-PASS

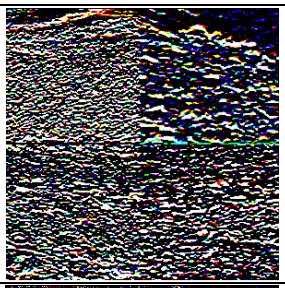
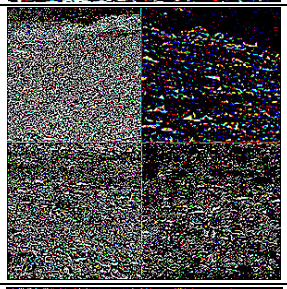
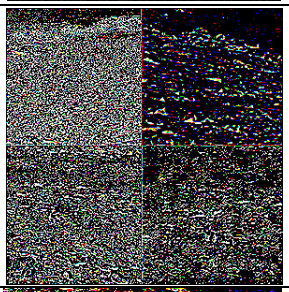
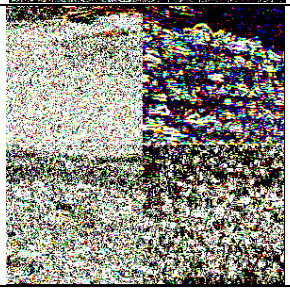
Here is the first filter, which will enable us to extract the high frequencies of a picture.

Techniques	Comment	Result
Naïve method : Edge detectror Prewit	We notice that the answer is very weak. It is difficult to distinguish a change of picture. The edges detected correspond to the outline of a rock or a stone, which is not the aim of our research. With sobel and compass [WEB01] we have similar results.	
Gaussien blur différence	If we use a simple gaussien difference on the original picture, the answer is very weak. It is useful to apply on the original picture a function of unsharping and then apply the gaussien difference. On this picture,we applies twice unsharping function.	
Laplacian convolution	Le laplacien of the picture gives us a quite satisfying answer.	
aaThreshold	The algorithm aaThreshold calculates the difference of value between two pixels. That is a rough approximation of the derivate from the slope between two adjacent pixels. For example, if two adjacent pixels have the same value, their slope is null. From this fact we fix a threshold in order to emphasize the most significant slopes. The advantage of this method, in regard to the 3 previous, is that it is dependent only of one of these neighbors. This allows us to stronger emphasize the variations of slope.	

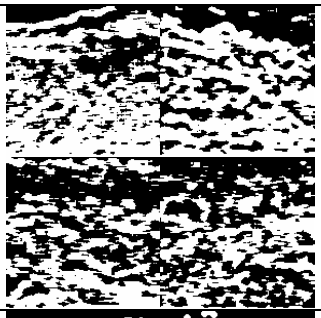
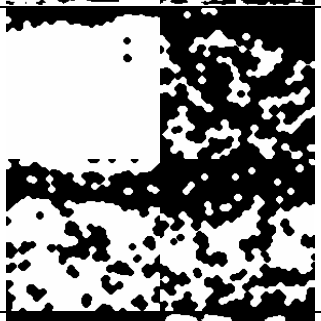
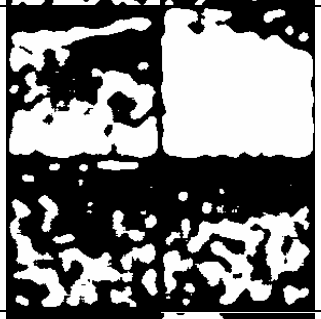
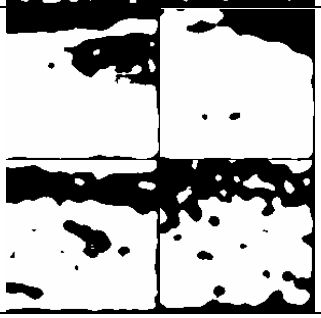
One has to apply to each one a threshold in order to point out the points, which interest us and eliminate the noise. The threshold value will be calculated from the result found in order to maximize the answer for each of the three channels. We will only show the values superior to threshold found (discretization).

3.1.4.2 STEP 2 : DISCRETIZATION

For each of the methods used, we have chosen a different way to maximize the output.

Techniques	Comment	Result
Naïve method : Edge detectror Prewit	Here the threshold is fixed to the average of the pixel values for each channel.	
Gaussien blur différence	The threshold here is fixed on half of the maximum value of the pixels for each channel.	
Laplacian convolution	Here the threshold is fixed on the average of the pixel values for each channel.	
aaThreshold	The threshold here is fixed at the median of the pixel values for each channel.	

3.1.4.3 STEP 3 : DATA SMOOTHING

Techniques	Comment	Result
Naïve method : Edge detectror Prewit	This method works well especially if one tries to extract the edge. In our case, they are not apparent; therefore it becomes very difficult to make a continuous surface because the obtained result corresponds to noise. ***Image processing :*** aaSmooth(9) : 2 times.	
Gaussien blur différence	We obtain a surface output only for the first surface, But unfortunately the occlusions are not detected. It was difficult to choose an efficient threshold in each case. ***Image processing :*** Morphological operators : dilatation (disc 4) and closure (disc 4)	
Laplacian convolution	This method being close to the latters, one could expect similar results in terms of efficiency. We notice that only big alterations of surface are recorded ***Image processing :*** aaSmooth(8) : 2 times.	
aaThreshold	It is from far the most relying method, because it gives us an output for each present situation. We obtain good quality closed surfaces, from which we will be able to extract the essential information. ***Image processing :*** aaSmooth(9) : 3 times.	

We choose aaThreshold, which gives us the best output by carrying out less operation. This way, we have distinctive blobs.

Now, we have to treat this information in order to make it useful. The first step is to calculate the surface (in pixel) of each blob, and then we will try to find out their centroïd. This way, we will be able to make a treatment for each of the blob and apply a list of conditions. For example:

Eliminate the blobs, which are under a certain size (in surface), eliminate the blobs, which lie at a certain position or distance…

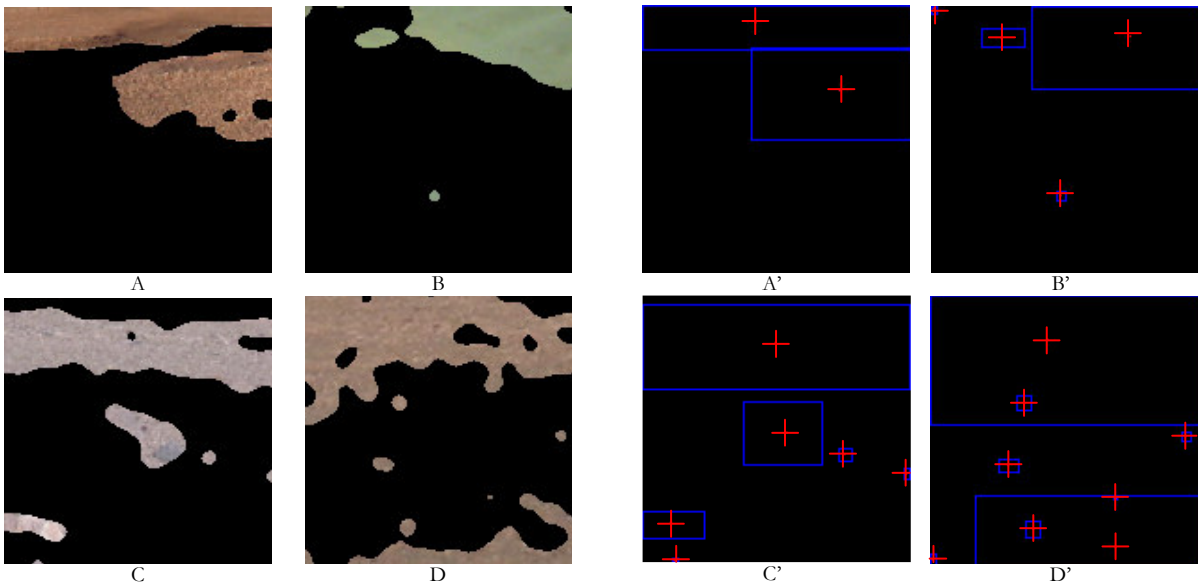


Figure 21 Left: Picture with the blob mask, Right: in blue min and max for each area, in red the centroïd.

Now that we have the information, we need to format them in order to discretize them so they can enter our « Segmentation Map ».

In each segment we will put information concerning the size, we will not fill in all the blank squares but just a continuous line, in which one will find many information concerning the same blob. For example we will have on the third image (Figure 21 C & C') as an output :

12348	12348	12348	12348	12348	12348		
					12348	12348	12348
			1054				
			1054				

Figure 22 Occlusion detector map, for each of the segment we put the value of the detected size of occlusion.

In order to have the ponderated Occlusion detector map we need to apply some rules. The final weight of each segment depends from the distance of each occlusion.

Weighting Table	
Infinite	100
Distant field 3	100
Distant field 2	100
Distant field 1	100
Far field 3	50
Far field 2	25
Far field 1	10
Mid field	0
Near field	0

Table 1 Weighting Tables for Occlusion detector map

3.1.5 Color detector

The color detector will just calculate the average of colors for a segment by eliminating abhorring values if there are any. The calculation of the average is made by researching the median of colors, which does not go over ½*π for Hue and 25% Saturation and Intensity (HSI system). If one or several colors move away, then the most distant one will be noted as "unexpected color". We use the median color because this is a detected color as output, not like the mean color that can be a interpolated color.

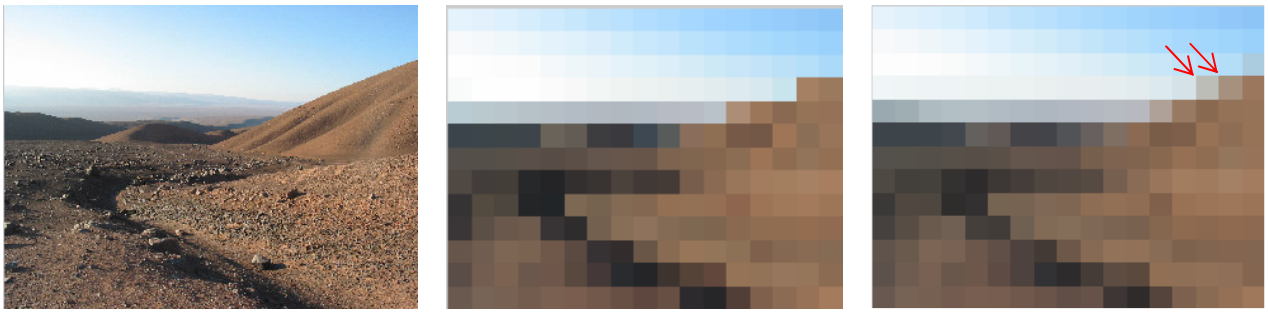


Figure 23 Left: Original picture, Center: Median Color Map, Right: Mean Color Map, interpolated color represented by red arrows.

3.1.6 Unique color detection

We have made this system in order to use this information as a science target. It can happen that on finds an herb blade in the desert, or a stone, which will have an usual color. It can be very interesting to stock these information for a further research.

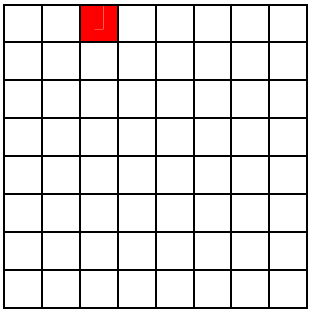


Figure 24 Left: Unexpected color detected, Right Unique color detection map

3.1.7 Blob detector – Rock detector

We are going to use the same proceeding that we used for the obstacle detector, which is the aaThreshold algorithm. But we are here going to modify the value of the threshold. This will enable us to distinguish other details, like stones, small rocks. Besides, this method has been implemented by the underbody rock detection.

We are going to use the rock detector to track objects lying at a distance between 10 to 50 [m]. As for the Occlusion detector we will store the surface in pixel of the stone. One can also afterwards store the rock's shape, the centroïd, the color, ex. .

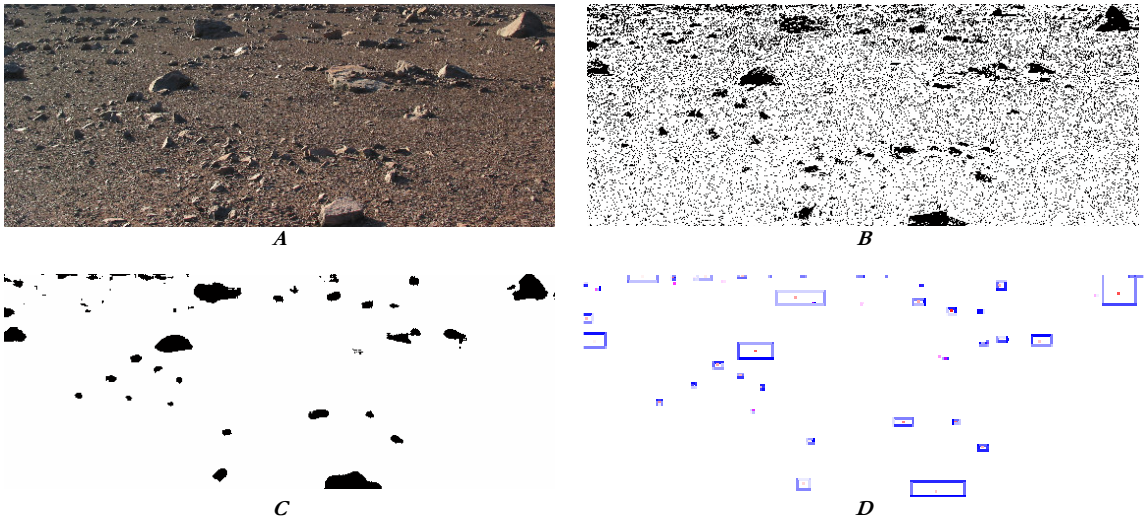


Figure 25 *A*: Original picture, *B*: aaThreshold algorithm, *C*: Smoothing, *D*: Blob detector

It can be useful for further science targets to know the density of the rocks for an area. It can also be likely to choose another path if the density of the rocks is too big and that we fear that damages can occur on the rover.

Results for the underbody rock detection see: Appendix A4 - Rock Detector for rover's underbody

3.1.8 Texture classification

The method that we are going to use here for detection and classification of texture is the Fast Fourier Transform. It is one of the most rapid methods to characterize a texture in a segmented picture.

The idea is to transform a picture which lies in the spatial area, in order to extract a picture, which is in domain of frequencies.

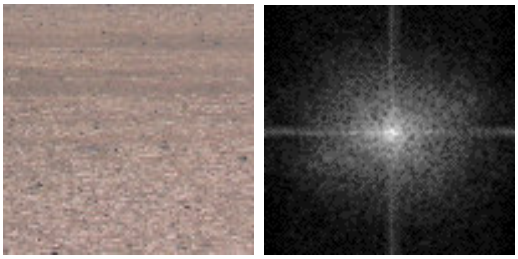


Figure 26 Left: original picture, Right: fast Fourier transform, (log(abs(fft)))

Once this operation is done, we can characterize this image by cutting it into areas of frequencies. To do so, we have cut the image into concentric circles with linear distances of radius and we calculate the value for each of the circles.

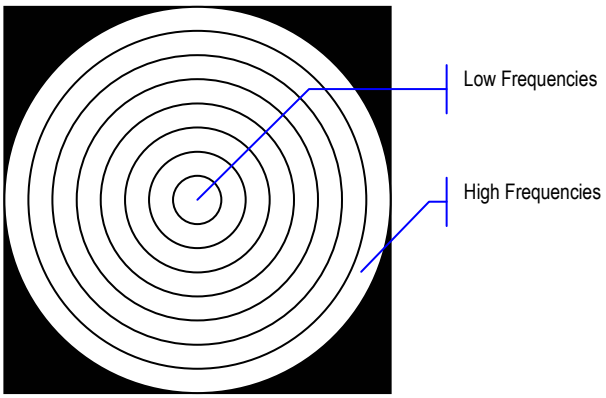


Figure 27 Fourier Transform frequencies representation, an example with 8 equivalent spaced radius.

This way, we create a vector with the value of each circle. This method will enable us to classify and compare by a simple manner a segment with another. We can also extract other information of a Fourier transform like main orientations. But in our case, this is not useful, because the principal information that we would need do not depend on orientation

In our case, we are going to use a segmentation of 32x32 pixel, and we have chosen 5 concentric circles of equivalent spaced radius (Figure 27).

In the following example, we chose to treat a whole picture, in order to clearly point out the different textures, but it is evident that in our case that we are only going to use the area, which lies between 0-50 [m].

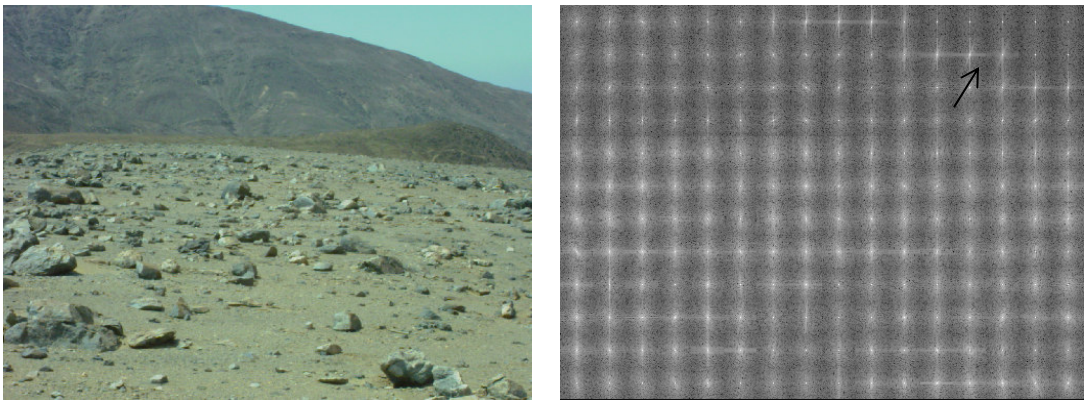


Figure 28 Left: original picture, Right: Fast Fourier Transform applied independently on each of the segement (32x32 pixels), Arrow: orientation detected.

3.1.9 Matching

For the matching we browse the whole picture of vectors Fourier transform and we compare it with the vectors that have already been indexed. If one of the values is different from a certain percentage *confl* (which is fixed her at 20%) relatively to the existing vectors, then this latter is added to the vectors database and the textures database.

In this example, we do not consider the color in the matching of textures; it is evident that this information is essential to make the matching.

One of the main ideas is to create a database well supplied with the textures that we have met in order to detect the changes of environments. All the textures, which are seen, are numbered and we also add a classification, which will enable us to tell if a texture is common or rare.

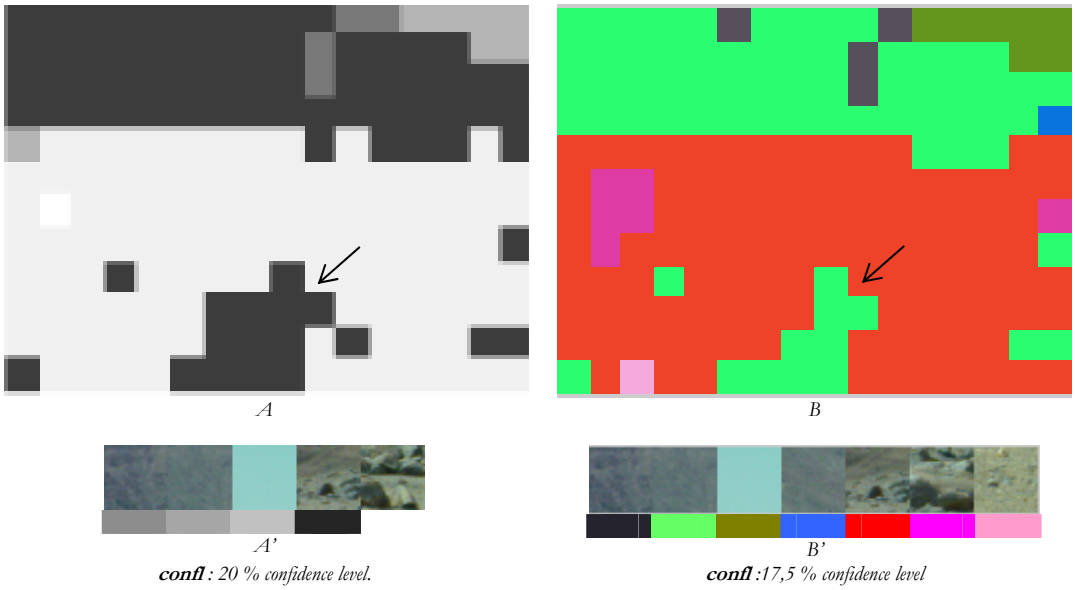


Figure 29 Examples of texture detection for a whole picture, *A* & *B* shows the texture equivalent areas, *A'* & *B'* shows generated texture vector for each picture. Original picture before processing in Figure 28 Left

We can see on Figure 29, textures that have been detected (on original picture Figure 29 Left) as having equivalent frequency proprieties. On both pictures, at the bottom in the middle (black arrow), we clearly see that there is a distinction between the sandy ground and the stony ground. Therefore we know that traversability is easier at the centre of the picture. In our example, the sandy ground unites with the mountains in the « Distant Field », which will not be the case afterwards because we are going to limit our detection by Far Field 3 (FF3: explanations lower in the text). (see other result in section **Error! Reference source not found.**)

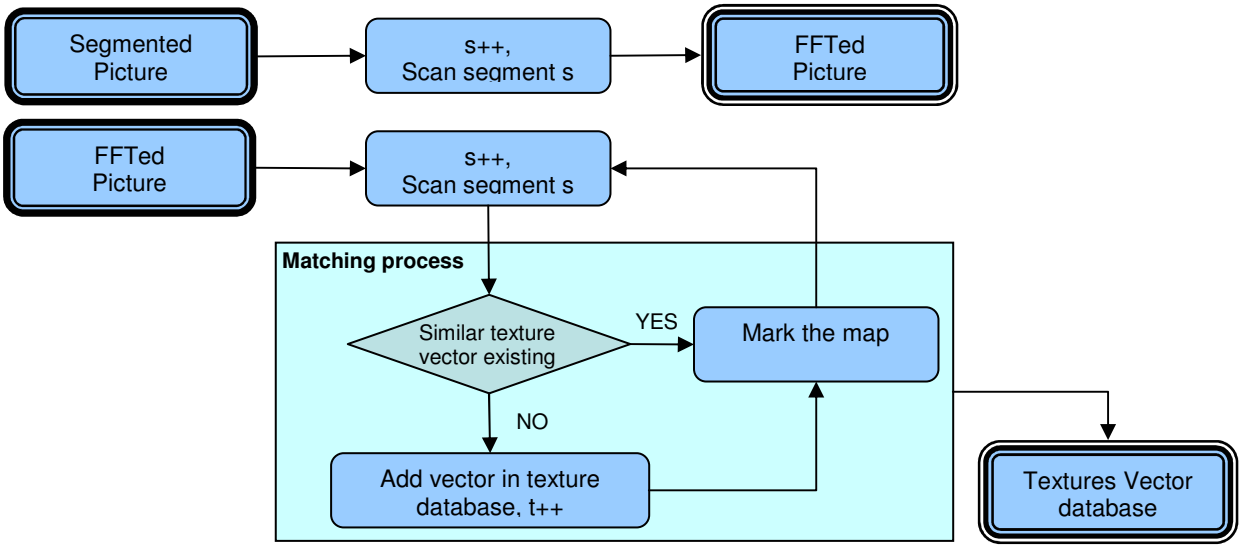


Figure 30 Texture classification process (algorithm aaFFTALL)

3.1.10 Distances Rules

Once we get the distance on a picture (3.2), we like to use this information and create rules with it. What do we mean by rules? Let's take a simple example: if we analyze a picture and we found that on the whole picture the most far away distance is at 30 [m], we can say that we face an obstacle and we have no information about what lies behind. Thus, we know that the traversability of the field will be rude, and it would be better to use another path.

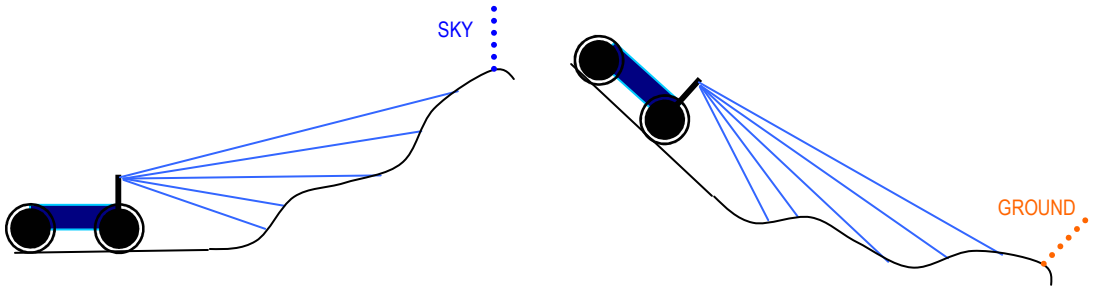


Figure 31 Field analysis examples, missing of information from a certain distance.

In order to complete this analysis, we have to discretize the distances (Figure 32), because we know we will not get continuous distance information on the picture (as if we could get one with the laser for example). In our case we have decided to segment distances with the non-linear ranges. We made this choice because of the perspectives of the picture and also because of the approximation errors that occurs during the long distance calculation, we suggest this segmentation:

From	To	Area's name
1	10	**Near field**
10	20	**Mid field**
20	30	**Far field 1**
30	50	**Far field 2**
50	100	**Far field 3**
100	300	**Distant field 1**
300	500	**Distant field 2**
500	1000	**Distant field 3**
1000	+∞	**Infinite**

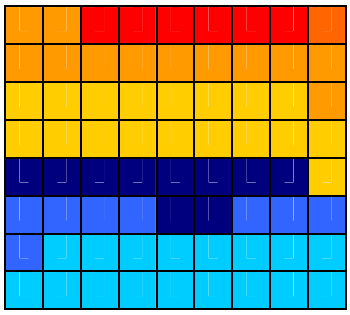


Figure 32 Show us the range of distances in meter . Moreover we will use this color convention for whole the project. Right: Generated Result of a distance map

3.1.10.1 Rule 1: Danger

The first condition that we set is the principal rule, which will serve the security of the rover. If there is a lack of information in at least one of the slices of the picture of Medium field to Far field 3 (10-100 [m]) and we have data's for the rest of the picture (distant field to infinite), there is a potential danger. The degree of danger will, of course, be relative to the distance at which it relies, a lack of information in the Medium field would be serious.

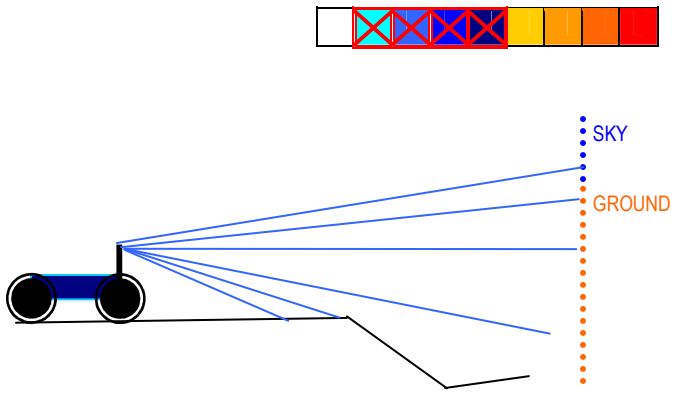


Figure 33 Top bar graphs: The red cross means that one (or more) of the area is missing

3.1.10.2 Rule 2: Obstacle

These and the following rules will help us to have a better understanding of the picture. If we face (close) a big obstacle, we will have continuous information until a certain point and after that nothing. In this type of cases, one must make arrangements in order to change the path.

Figure 34 White crosses means, obstacles starts from mid field or far field 1 or 2 or 3

3.1.10.3 Rule 3a: Rover go up

If we have information from the mid field to far field 3, and then nothing until infinite, we can conclude that the rover has a big probability to be on a rising slope.

3.1.10.4 Rule 3b: Rover go down

If, on the contrary, we have all the information until distant Field and nothing at infinite we can tell that we are on a descent. This can also be verified, if needed with the sky detector, which will tell us if it is the case (NO SKY DETECTED).

To succeed in treating the whole picture, in an efficient manner, we are going to use (as previously) the segmentation of the picture and write on this map the measured distances. Then, we will browse the map from top to bottom (vertically) to seek if some areas are not represented. If this is the case, we note the abrupt transition and remain vigilant about this part of the picture.

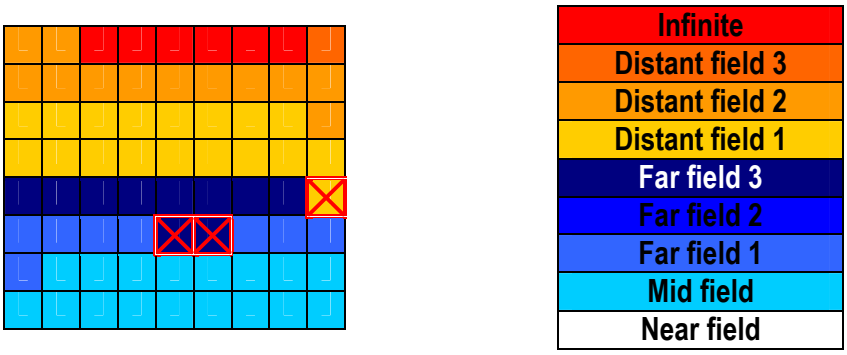


Figure 35 Left: Distances map: 3 discontinuites where discovered.

We use a weighting system to make a graduation of the danger. The rule are showed in below in the Table 2. The first segment scaned (from bottom to top) is in the first column and the second segment is in the first row. So for example we look at the example in Figure 35, we checked 3 segment: there is two segement with 100% of danger and one with 80% of danger.

Weighting Table								
Infinite								
Distant field 3	0							
Distant field 2	0	0						
Distant field 1	0	0	0					
Far field 3	25	25	25	0				
Far field 2	50	50	50	50	0			
Far field 1	75	75	75	75	75	0		
Mid field	100	100	100	100	100	100	0	
Near field	100	100	100	100	100	100	100	0

Table 2 Weighting for Distance rule map. In percent.

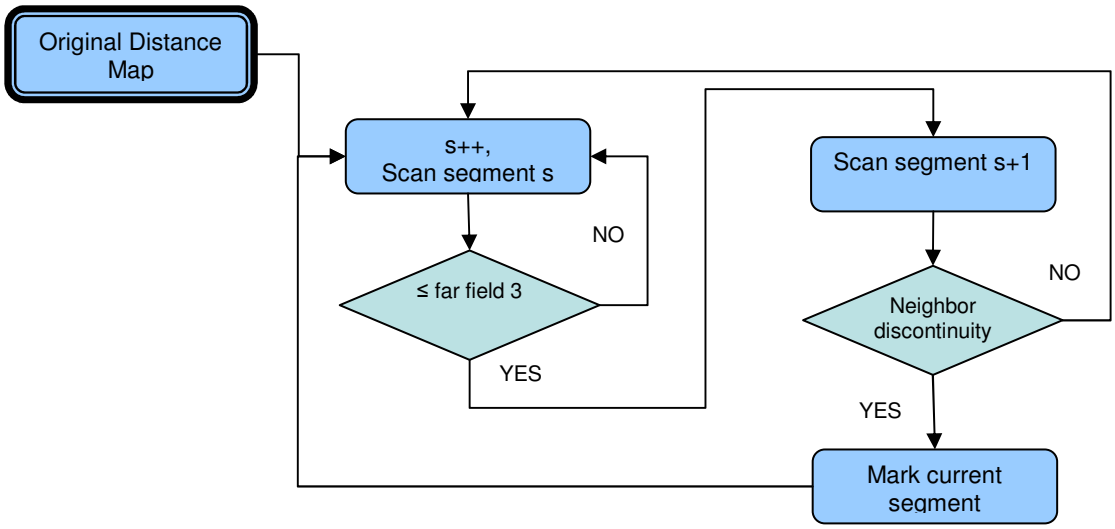


Figure 36 Distances rules map flowchart, s : position of the current segment

3.2 Part II : Geometric

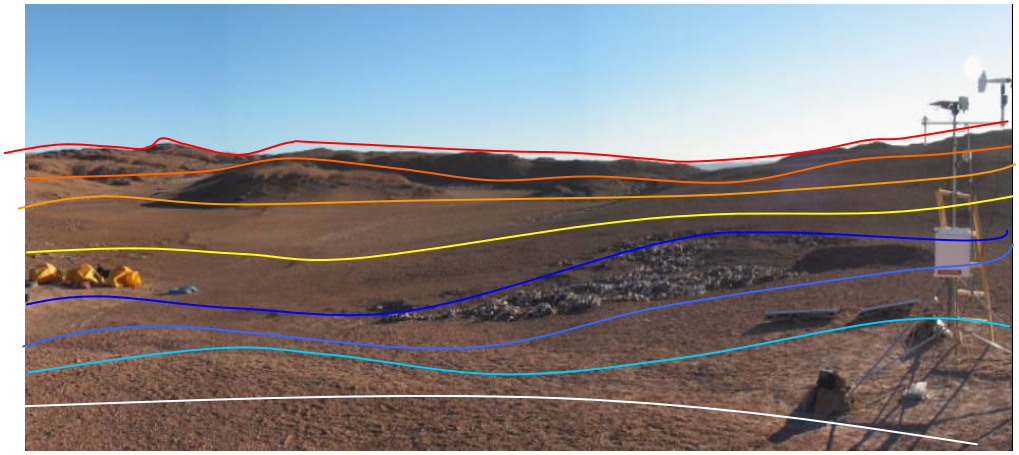


Figure 37 Geometrical equidistant segmentation

3.2.1 Introduction

It is essential in this kind of space to make extractions of distances. To know if a part of the picture that we analyze is close or distant.

We have at our disposal a laser and two cameras (as described above). The use of the laser is very interesting because it enables us to extract in a very precise manner and with a big density of information, in a range of 0 to 25-30 [m]

The disadvantage of the laser is that one cans make a linear scan (1D). One should have constructed a system (active or passive), which could have made the laser pivot in order to obtain scans in 2D. There is a main disadvantage to this: if the laser browsing is active, it increases considerably the consumption of energy. Moreover, being active or passive, a 1D or 2D will never give us enough information about our environment. We can have a very precise measurement of distances but for a very limited distance. And if we want to work with textures, one should make anyway an interpolation of the distances measured by the laser, in order to find their correspondence on the picture. This implies calibration procedures and additional treatments of pictures.

Therefore, we decided just to work with the stereo for the geometric part. The navigation system "near field" already uses the SVS software to make stereo [CUR031] [KKO991]. Therefore, it is evident to go on the same way by using SVS for the far field navigation. This way we could get a better resolution quality (as we are going to see it lower in the text Figure 56) We have decided to increase the baseline to 0.600 [m].

If one refers to [KKO992], the advised baseline is about 3-8 inches (0.076 – 0.203 [m]) if we want to get good result.

As we will also deduct by [CGR03] [CFOL00] the range resolution goes up (gets worse) as the square of the range. The baseline and focal length both have an inverse influence on the resolution, so that larger baselines and focal lengths make the range resolution better.

Knowing these constraints, we have nevertheless tried by several manners to calibrate the cameras (SVS Calibration Tools, Matlab Calibration Toolbox [W03]). The process of calibration was very long and gave us only poor results.

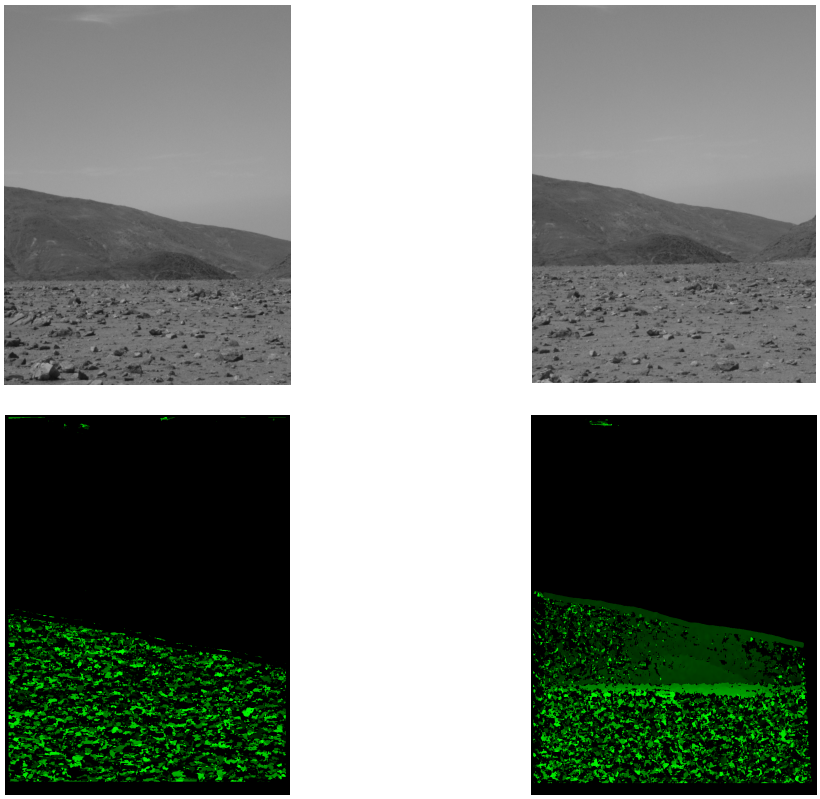


Figure 38 Top: Stereo picture with 0.600 [m] wide baseline,
Bottom disparity map, left: calibration attempt 1, right: calibration attempt 2

On Figure 38 at the bottom on the right, we can notice the best obtained result with the SVS calibration tools. On a SVS disparity map the lacking of depth information is represented by the black areas. A good disparity map should have smooth colored areas from light color's, which means near, to dark, color which mean far. We can clearly see through the disparity map that the output gives us only noise and in the best case we can notice a distant surface area; but as for the mid field and far field areas, the information are useless.

Other attempts of calibration have been made with no more success. At this stage, we hade to take a decision: Go on with calibration the cameras and use SVS? Or choose

software (i.e. Open CV [OCVRM])? Or try to develop an application dedicated to the far field navigation.

Go on in calibrating the cameras were out of question considering the time we spent on. Moreover, we can hardly imagine to recalibrate the cameras in the same circumstances especially if one stands in the desert (on the field). The choice of similar software was also put aside because even if we deal with a more efficient software, the process of calibration will be similar. Therefore, we chose to start over on new basis and develop a simple stereo system who will respect these two conditions:

- A simple calibration system, even auto-calibration.
- No limitation for the size of the baseline.

3.2.2 Choices and implementation

3.2.2.1 Goal

In this part of the project we will try to estimate the distances in a scene, in order to know if an object is closer than another or to evaluate the size of an object, being either a stone or a mountain.

3.2.2.2 Introduction

For this purpose, we are going to use stereoscopic vision. This method is based on the acquisition of images resulting from two cameras, which focus on the same scene at the same moment (another method, which consist by only one moving camera exists also). The distance between the two cameras is steady and we will call it the "baseline". If one looks closer to the two pictures, one will notice that some objects have different position on the two pictures. The distance separating the two representations of the object on the picture is called « disparity ». Disparity is directly related to the distance of the object in the scene. The further the object is away, the smaller the disparity, and it will tend to zero if the object lies at infinity.

3.2.2.3 Methods

The method that we are going to discuss here is adapted to the « far field navigation ». It means that it is very well adapted for evaluation or representation of big spaces, for which the required depth-of-field is large. The simplicity of the method relies also on the fact that distant objects have a small disparity. And it is on this property that we had the idea of calibrating the cameras (the definition of calibration will be given later on). But before being able to gage, we have to use a method, which will enable us to say with a very good accuracy, which part in the picture corresponds to the same one in the other picture. In order to achieve this aim, we are going to use the SIFT algorithm (Scale Invariant Feature Transform).

3.2.3 Introduction to SIFT

SIFT is a very powerful algorithm, which has been developed by David G. Lowe [DGL99] and has been released for the first time very recently (June 2003).The capacity of this algorithm, is that first of all it is able to find objects in a scene, which have been submitted to, for example, rotations, translations and/or distortions and partly lightening change. Previous approaches to local feature generation lacked invariance to scale and were more sensitive to projective distortion and illumination change. In our case we will only use the rotation and translation for the calibration of the camera and for disparity we will use translation and scaling, the deformation case could also interest us if we used cameras with different optics. In this project, we will only discuss the case of two cameras which have the same properties.

The scale-invariant features are efficiently identified by using a processed filtering approach. The process is to identify key locations in scale space by looking for locations that are maxima or minima of a difference-of-Gaussian function. The Gaussian kernel and its derivatives are the only possible smoothing kernels for scale space analysis and also have a high level of efficiency for the rotation invariance. Each point is used to generate a feature vector that describes the local image region sampled relative to its scale-space coordinate frame. The features achieve partial invariance to local variations, such as affine or 3D projections, by blurring image gradient locations. The resulting feature vectors are called SIFT **keypoints**

SIFT, as described in [DGL04], consists of four major stages:

(1) scale-space peak selection

Potential interest points are identified by scanning the image over location and scale. This is implemented efficiently by constructing a Gaussian pyramid and searching for local peaks (keypoints) in a series of difference-of-Gaussian (DoG) images.

(2) keypoint localization

Candidate keypoints are localized to sub-pixel accuracy and eliminated if found to be unstable.

(3) orientation assignment

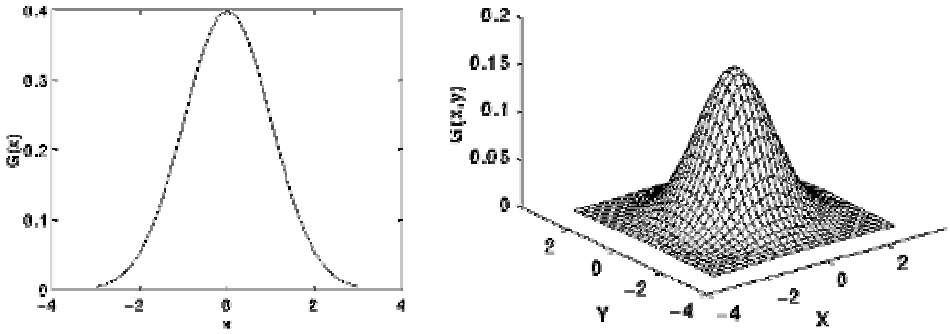
identifies the dominant orientations for each keypoint based on its local image patch. The assigned orientation(s), scale and location for each keypoint enables SIFT to construct a canonical view for the keypoint that is invariant to similarity transforms

(4) keypoint descriptor.

local image descriptor for each keypoint, based upon the image gradients in its local neighborhood

(1) The Gaussian smoothing operator is a 2D convolution operator (figure c) that is used to blur images and remove detail and noise. The 2D Gaussian function is separable; by applying two passes of the 1D Gaussian function in the horizontal and vertical directions.

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$



A *B* *C*

Figure 39 Gaussian Representation

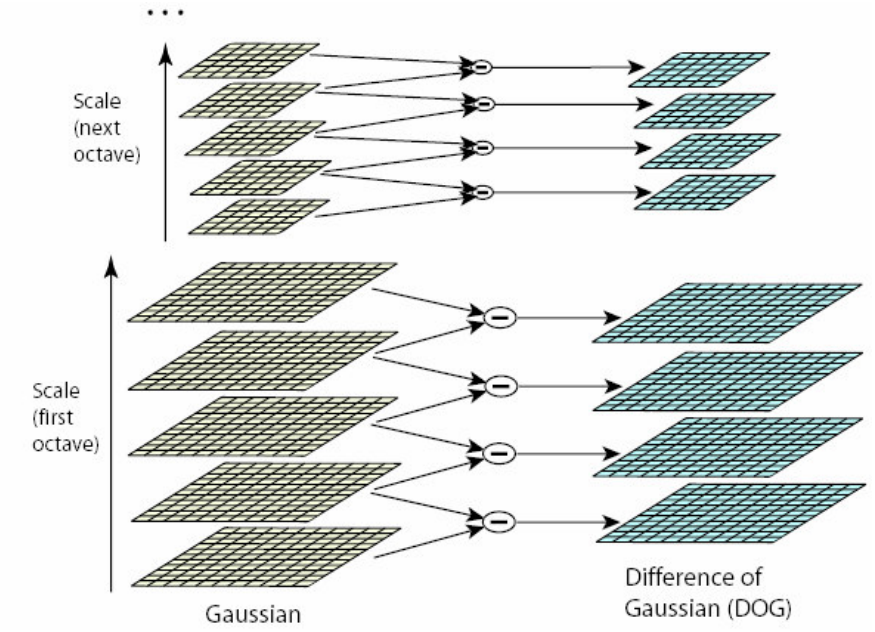
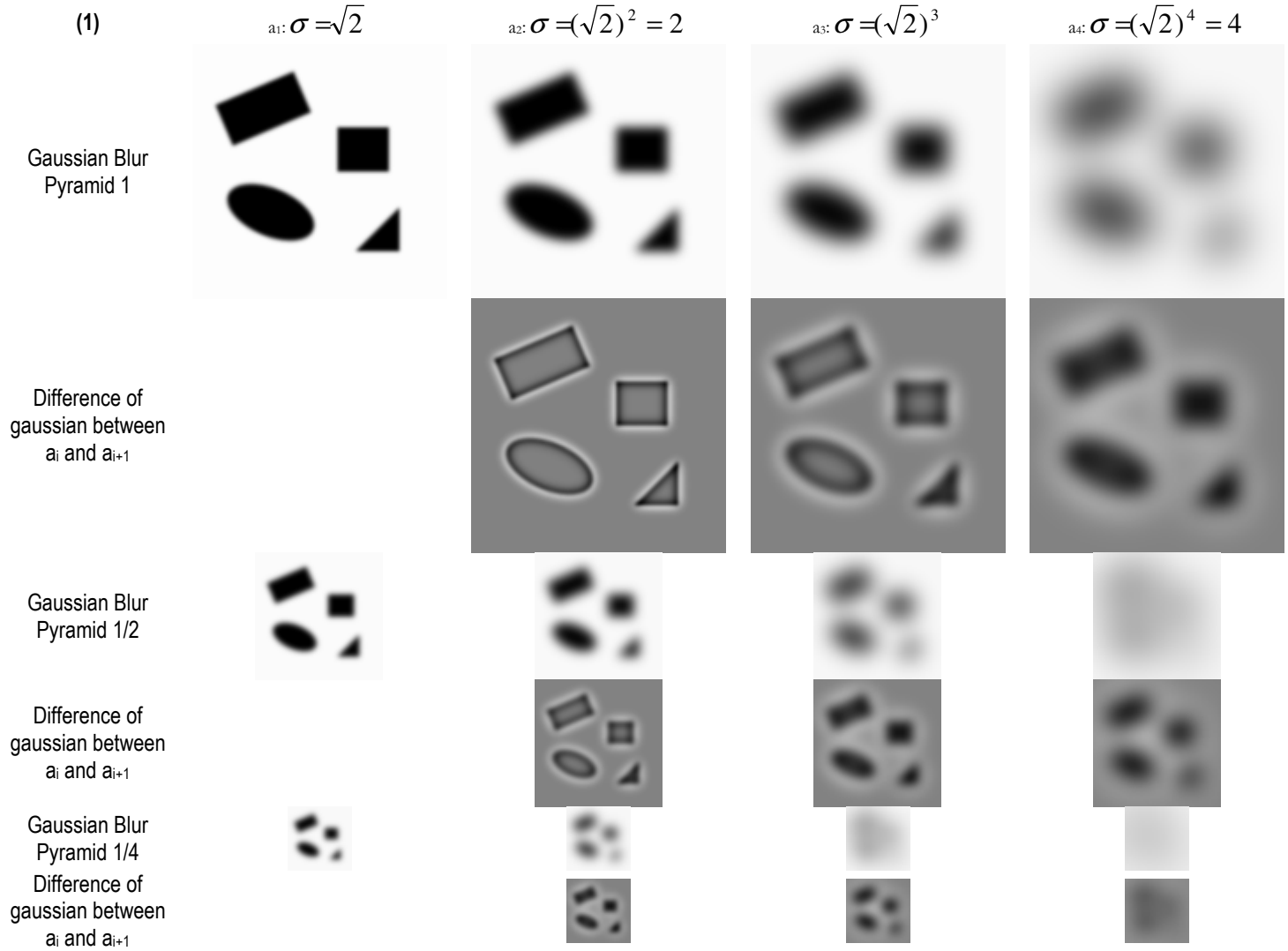


Figure 40 Gaussian pyramids

We are now going to illustrate each of the stages of SIFT in the following example. The chosen image is a very simple example and creates very few keypoints for a better understanding.

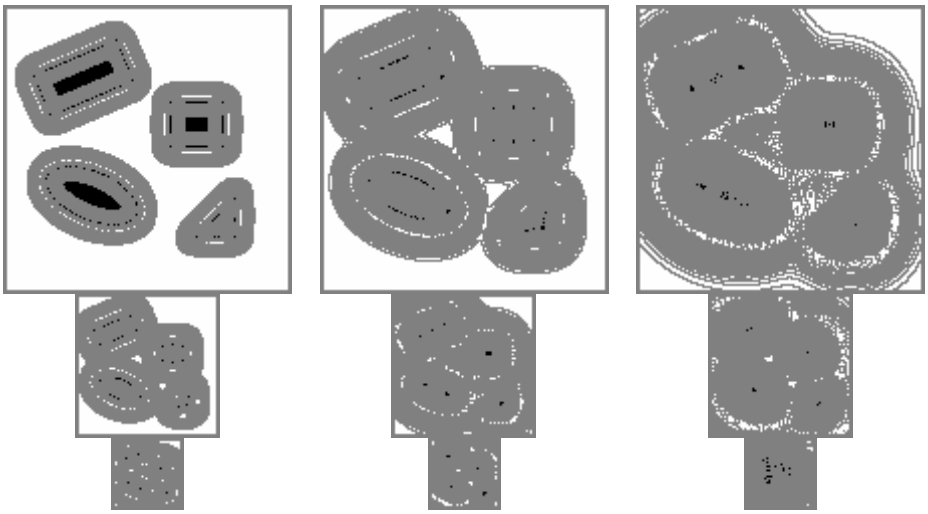
Figure 41 Original Picture

(1) here is the illustration of point 1. On The representation of the Difference of Gaussian pictures, the grey color means 0, all the minima are represented in black and the maxima in White.

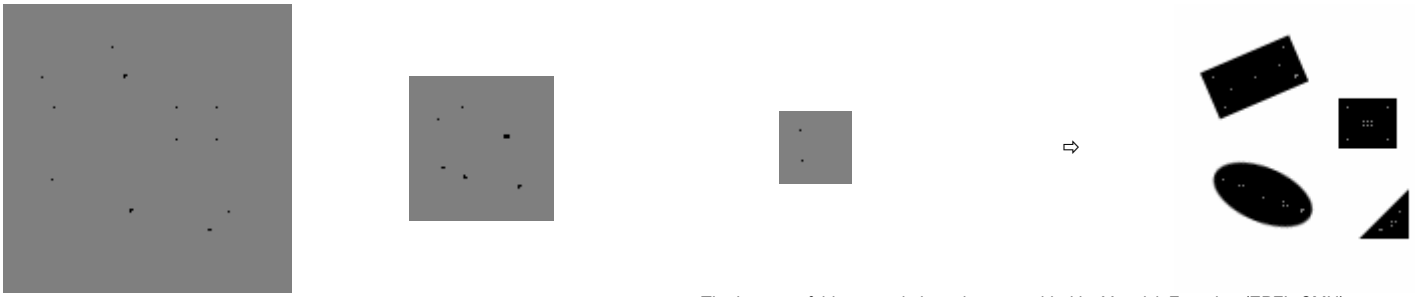


(2) Here we a making a Thresholding in order to extract the maxima and the minima in the picture

(2)



The only best results will be kept at every octave of the pyramid. (3) Then an description vector will be determined for each of the keypoints (for more information about the description vector [DGL03]).



The images of this example have been provided by Yannick Fournier. (EPFL-CMU)

3.2.4 Matching

The best candidate match for each keypoint is found by identifying its nearest neighbor in the database of keypoints from training images. The nearest neighbor is defined as the keypoint with minimum Euclidean distance for the invariant descriptor vector as described.

3.2.5 Improvement

We have done here a few modifications in order to adapt as best as possible the algorithm to stereoscopy. In our case we know that if the cameras are calibrated, the corresponding keypoints have a strong probability to lie on the same epipolar. That is why it has been decided to make the matching in sub-zones of the image. The image is then divided into many pieces overlapping one another. The overlapping is important for diverse reasons. One hand this method prevents from loosing the keypoints which would be in two part of a picture at the same time, and on the other hand the SIFT algorithm does not detect the keypoints at the edges of the image. Because the edges of the image are considered as instable. The image is divided into segments, which are parallel to the baseline in order to guaranty that a point lying on the same epipolar appears on the segment.

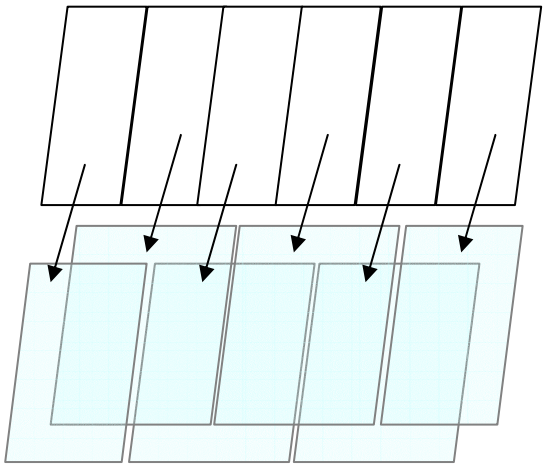


Figure 42 Example of keypoints areas segmentation for a vertical baseline. The white top part is the segmented picture in vertical band and the blue areas are the several matching segment.

The overlapping reaches half of the neighboring bands (as we can see in Figure 42). This gives us a size of matching area which is the double of a band. Finally, a sorting is carried out in order to eliminate the ones, which would be represented several times in the keypoints list.

Several measures have been carried out in order to find a segmentation, which would be adequate for the kind of picture we have in this project (clear and flat spaces). The orientation of the bands is directly linked with the orientation of the baseline. In our example we have to deal with a vertical baseline, thus the segments have the same positions as on Figure 42.

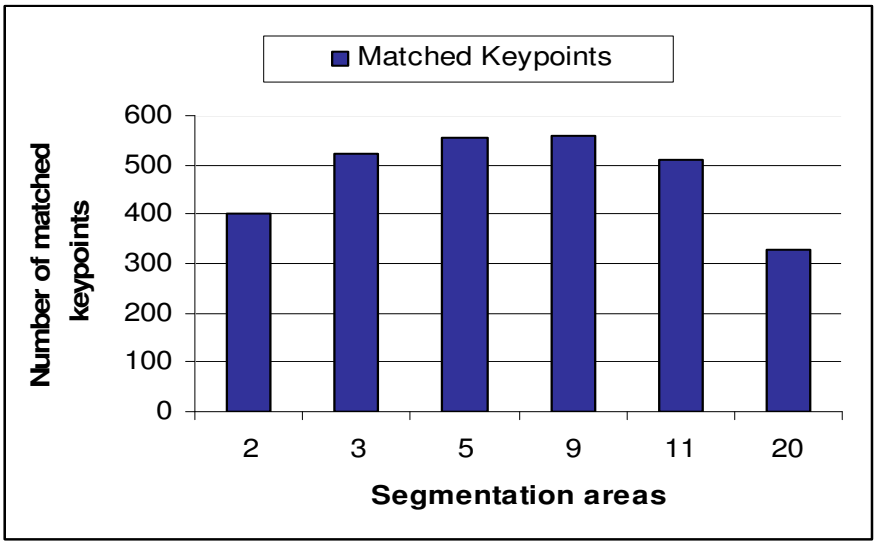


Figure 43 Matched keypoint vs number of image vertical segmentation

We made our measures (see Figure 42) in order to respect a rate of false match inferior to 1%.. We can notice that if the segmentation area is 2, the number of match is poor. This can be explained for one given point (on the first image), the matching algorithm is going to seek on the whole of existing points on the second picture. (N.B. A segmentation area of size 2 is like taking the whole surface, because we always take in consideration the neighboring segment, which enables us to make an overlapping) Consequently the rate of false match is higher and the number of match is inferior

On the contrary, if we have a look at the results for 20 segments, we will notice that the matching looses also its efficiency. This is clearly explained by the much closed objects (mid field). The points lying on the two segments are different (i.e. non-adjacent); i.e. the ones which have a disparity over 3 times the size of the width of a band segment will never be matched. This results clearly in a lack of information in the near field.

Consequently we decided to work, for the continuation of the project, with a segmentation area of 9 bands.

3.2.6 The Stereo calibration

After having set the cameras and stabilized their orientation (rotation XYZ and the baseline), the procedure of calibration is made. This operation is carried out only once. The main idea of calibration is to extract the extrinsic parameters of the cameras and also add intrinsic parameters, which are known by the physical propriety of the cameras. These parameters will enable us in some way to rectify the cameras in order to have a view of the scene, which will be as identical as possible for the two pictures. With the ordinary calibration methods, the cameras are trained with targets, that both cameras can see [KKO991]. This procedure can be long and requires the intervention of an external system, but this last method has the advantage to give us intrinsic and extrinsic parameters of the stereo system with a big accuracy, e.g. the image distortion.

In our case we are going to calculate some extrinsic parameters such as the image rotation or the translation and provide to the system intrinsic parameters: such as the baseline, the focal length of the optical, the size for the picture, the real size of the pixels, etc; roughly speaking all the parameters known to be invariable, which will not suffer modifications whatever the situation is.

The algorithm used here will enable us to look for the most distant points in order to calibrate the picture. In our case we know that the points lying at a faraway distance (more than one kilometer) are stable enough (Figure 56) to enable us to undertake this operation. If the calibration is calculated with points being at a distance lower than this limit, the quality of the extraction of distance in the image will be strongly faded. We use for the calibration the point which is lower than the resolution of the stereo vision and the resolution of each camera. We will discuss these points in 3.2.9.1.

For this task, we are going to use the SkyDetector algorithm (Figure 44), which will give us all the proprieties that interest us to extract the zone in which lies the skyline.

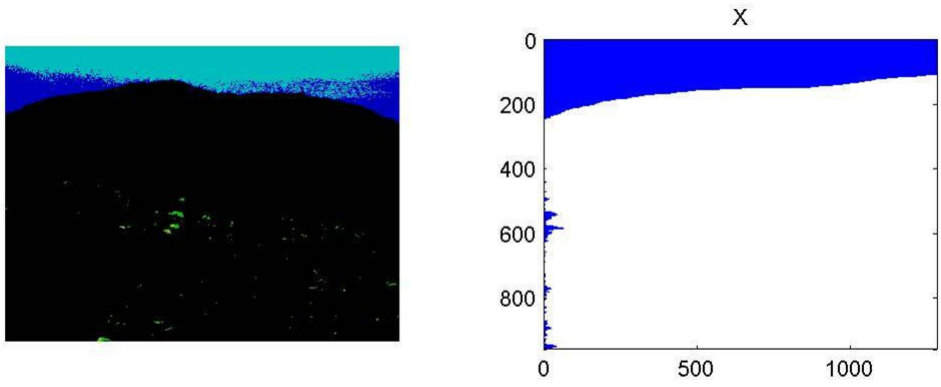


Figure 44 Sky Detector

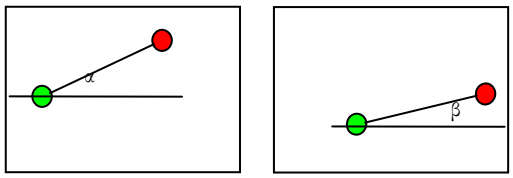
This operation will be carried out on both stereo pictures at the same time, in this operation we are going to extract the zone of the picture, which includes the sky and the skyline (Figure 45).

Figure 45 Left : Left stereo sky picture, Right: Right stereo sky picture

We will notice that the size (in height) of the picture is not the same as well as the position of the mountain on the picture, which is normal. And that is precisely to achieve this aim that we are going to carry out the calibration.

From the two pictures, we are going to calculate the keypoints. Each one of the two images will give us a different number of keypoints, which we will need later to match. Once this operation is set, we find ourselves with very stable reference points on both images.

From now on, we can calculate the Euclidian distance between the two points lying on the two pictures. We will use statistical methods to ensure that distances are high quality and also to reject the aberrant values. This is why we will throw away the values being more or less than 2,5 % of the median value of the distances (quantile 95%). And if the thrown away number is above 10%, the pictures of the calibration are then considered as bad.



$$\alpha_u = \arctan\left(\frac{I_{\alpha xu} - I_{\alpha x(u+1)}}{I_{\alpha yu} - I_{\alpha y(u+1)}}\right)$$

$$\beta_u = \arctan\left(\frac{I_{\beta xu} - I_{\beta x(u+1)}}{I_{\beta yu} - I_{\beta y(u+1)}}\right)$$

$$\gamma_u = \alpha_u - \beta_u$$

Figure 46 Green: dot on left picture, Reg dot on right picture

To correct the rotation we take the slope between two points on the same picture (*I*) and calculate the difference between them. This operation is carried out for all the pair of points (*u* and *u+1*). We apply the same statistical methods used previously. And the γ median will give us the slope.

As soon as these two operations are achieved with success, we get a calibration file, in which one will find the intrinsic and extrinsic values of the stereo system.

The calibration file contains theses information's:

Mt	Translation Matrix in x and y [pixel] (float)
Zrot	Rotation value γ_u [deg] (float)
f	Focal length for both camera's [m] (float)
p	Pixel Size in x and y for both camera's [m] (float)
b	The camera's baseline [m] (float)
SR	Rectification matrix (int)
OS	Original Picture size in x and y [pixel] (int)

There are, however, conditions to respect to have a calibration of good quality. If the horizon is flat and monotonous, the calibration will be difficult. In the same way, to make a successful calibration the rover must have a view of the skyline, and this one must be distant (more than 1 [km], see 3.2.9.1). We choose this type of simple calibration, because it is exactly appropriate for the needs of the extraction of distance. The second reason that motivates our choice is the self calibration. It is clear that with this method, the extrinsic calculation of parameters like the lens distortion is not possible. This is why that we provide in advance the maximum number of intrinsic parameters and make a simplified correction of the lens distortion (see 3.2.10).

3.2.6.1 Results

3.2.6.1.1 Translation Vector

Checking the quality of calibration is a quite difficult exercise, because distances are big. However, we chose a simple method, which shows to be efficient to reach our goal. A series of 10 stereo pairs have been chosen to test the quality of calibration. The pairs of pictures, which have been chosen are almost parallel, the angle according to the Z axis is inferior to 1 degree. We have compared the translation values calculated by calibration with pixels with selected pixel's lying on each stereo pairs. The results are represented on Figure 47.

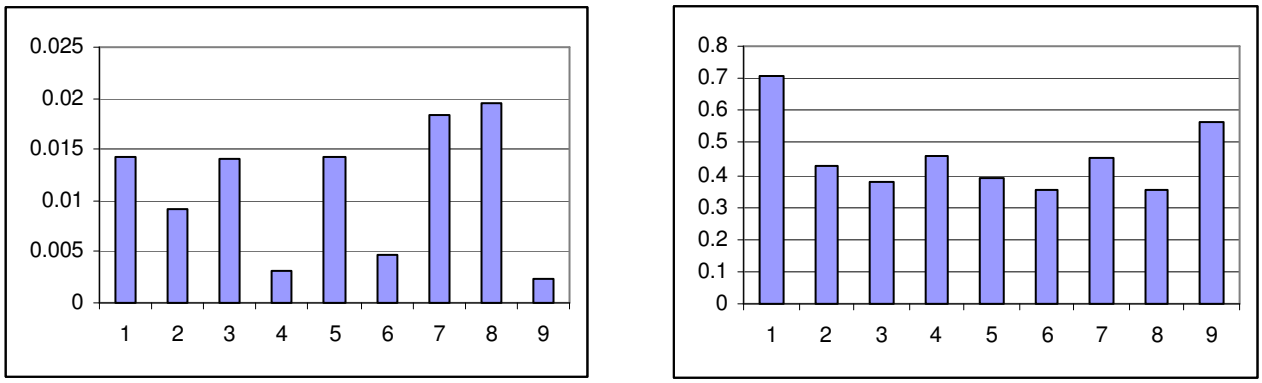


Figure 47 Left: computed error/measured error in % vs number of stereo pairs, Right: Measured angles for each stereo pairs

Out of the 10 pictures tested, only one picture was not accepted because it did not contain enough keypoints. On average 8.4 keypoint was found for each picture. The robustness of keypoints for calibration is much more higher than the one for the research of distances. The parameters are more restrictive, which explains better the number of keypoints detected. We calculated with the **Error! Reference source not found.** Left, the error average, which is of 1.19% for deviation of the translation position by pixel. This value is in reality even inferior, because in our pixel pointing technique we do not consider the sub pixels whereas SIFT does it. **Error! Reference source not found.** Right shows us the angles of rotation depending on the Z axis of each of the pairs of pictures. We can see that rotation is for the majority of the points inferior to 0.5 degree.

3.2.6.1.2 Z Rotation value

For rotation, the test is much simpler. We took a series of pictures, to which we have voluntarily given known angles and then measured the difference with the calculated angle during the camera calibration.

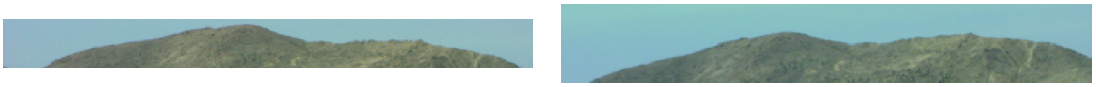


Figure 48 Detected Sky, Left: original picture, Right modified picture with a 5-degree angle rotation

On the example of Figure 49, an angle of 5 degree was given for the second picture. The calculated angle, with this method of calibration, is of 4.98 degree.

Here are below several other measures executed:

Angle rotation	Result by calibration	Error average
-10.00°	-10.00°	0.01%
5.00°	4.98°	0.39%
10.00°	10.02°	0.16%
20.00°	20.03°	0.13%
45.00°	44.94°	0.14%

Table 3 Result for several angle rotation computed by calibration

The error measured here is on average 0.17% of the original angle. The most important error occurs for the 5° angle. This can be explained by the bad quality of the algorithm of the drawing software (rotation non-smooth), which makes a rotation of the original picture. Therefore, we can conclude that the method detects in a very good way an angle, which would be applied to one of the two cameras in a non voluntary way.

3.2.7 Space evaluation

Here we are reaching the main purpose of our project, which is evaluation of distances. The first objective in our case is not to recreate a 3D map; but rather to "pick" the points in order to estimate the distances relating to a zone of the picture, in which we would be interested. Our purpose is to be able to see if there are, e.g. crevasses, excavations or ravines, which could not be detected by satellite (because of its resolution weakness), and that one can be estimated thank to the lack of information in a certain bracket of distance (as we can see it on the following illustration).

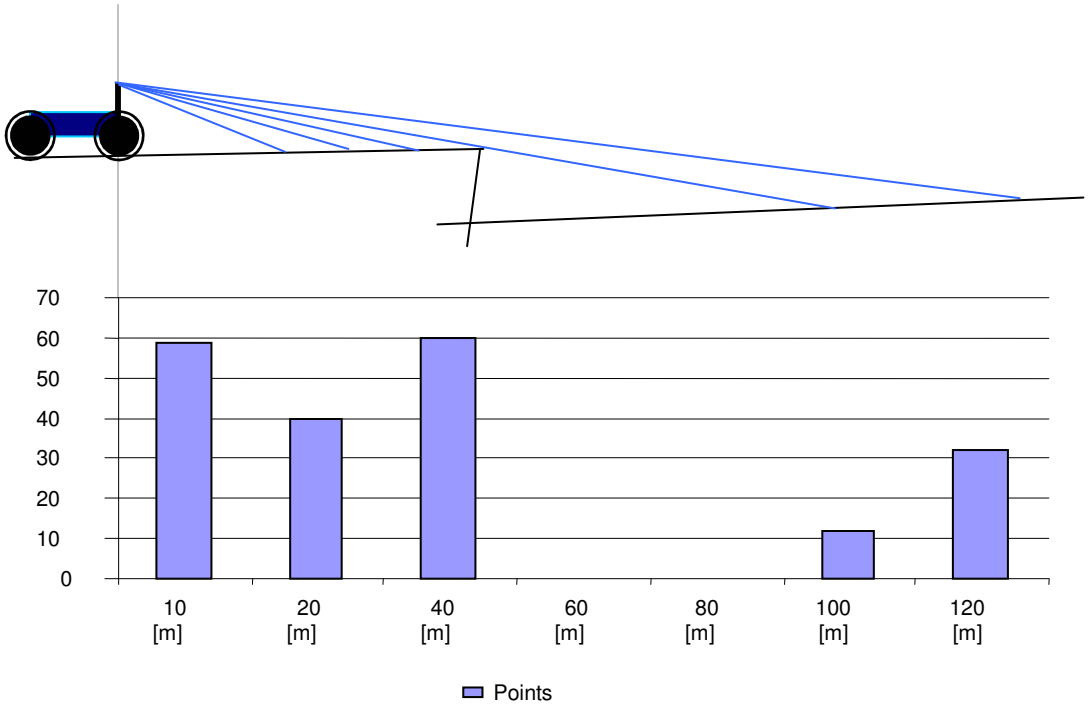


Figure 49 Example of number of detected points vs the distances. We can see a depression between 60 and 80 [m] by the lack of information

3.2.8 Picture cropping

The first operation carried out after the calculation of the calibration parameters is the application of these parameters to each one of the entering pictures. We will call this operation the « cropping » because in our case it concerns only the translation and the rotation of one image in comparison to another. The "cropping" will apply to each pair of entering pictures, acting a bit like a filter. It will be able to find the zone of vision, which is common to both images and will remodel the two original pictures in order to create two new ones, which will be calibrated. There are roughly 4 general configurations in case the cameras would have the same proprieties (same size of picture, same optics,…). The information which lies on the edges will simply be eliminated in order to prevent bad keypoints match.

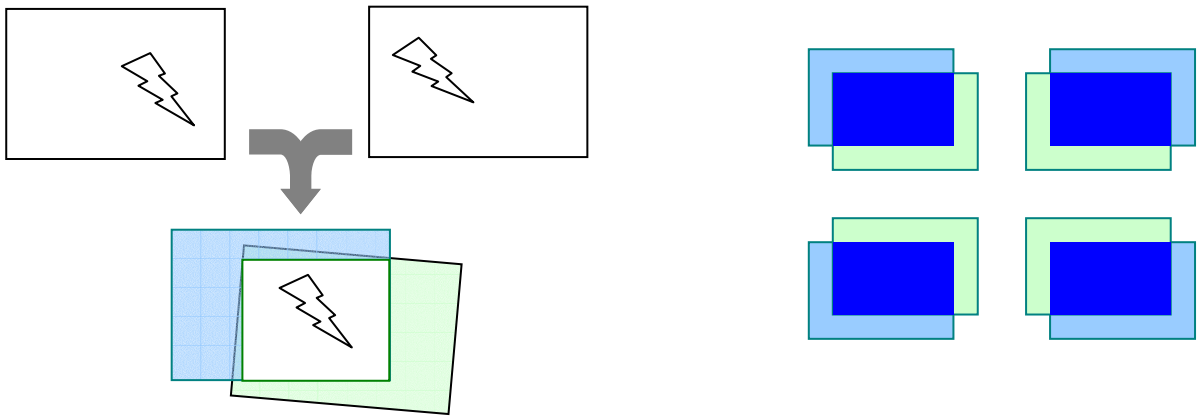


Figure 50 Left: picture which undergoes a rotation and translations, Right: in dark blue common pictures areas

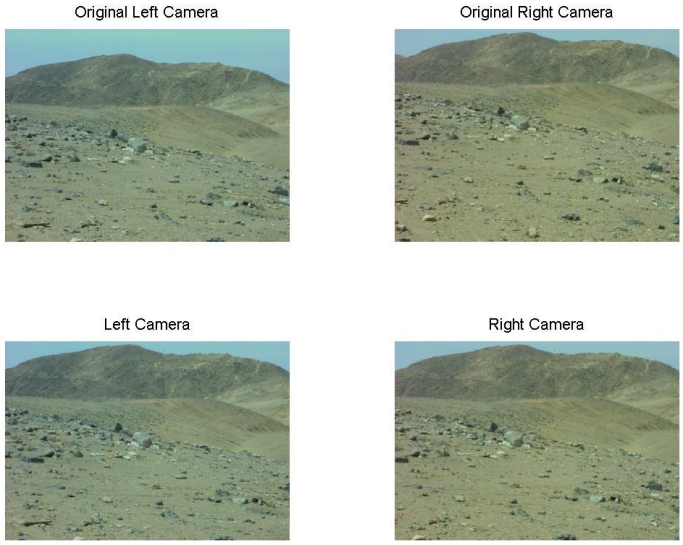


Figure 51 Result on field pictures Top: Before calibration, Bottom: After calibration.

3.2.9 Distances extraction

The calculation of the distance (in depth Z) between the camera and the point to be fixed is made according to the formulas expressed in [CHG03].

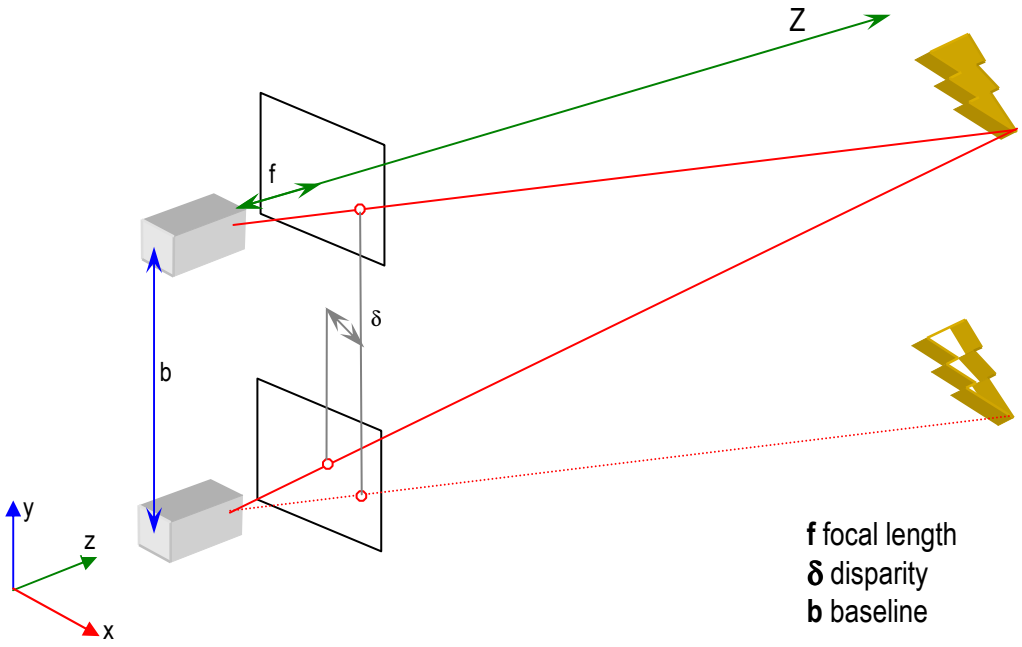


Figure 52 Regular stereo geometric illustration

We consider in our case that δ[x,y] is the disparity on the picture. We can also notice that the advantage to use the keypoints method is that it generates pixels positions, which are in float. This gives us a better accuracy in comparison to the conventional systems, which only have half pixel accuracy. The calculation of real coordinates of the points (X ;Y) from their projection on the picture is carried out in the following way: A unique referential and common to both images must be chosen, and the center (on X and Y) of the calibrated picture has been chosen ; the rest of the distances will be calculated.

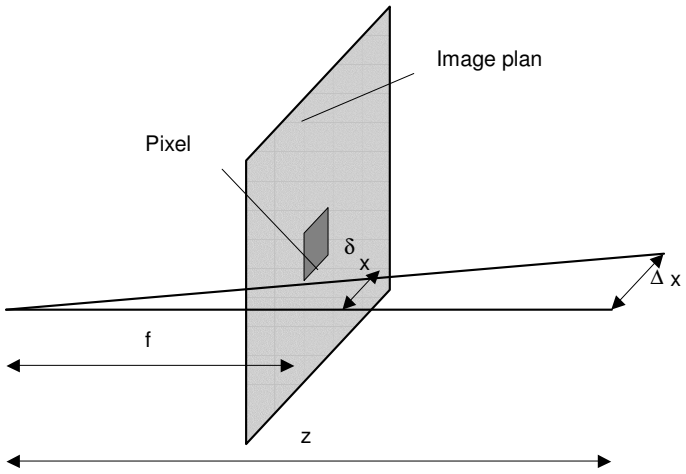


Figure 53 Geometrical relation between resolution Δx, the distance z, and camera's parameters δ and f.

$$\begin{pmatrix}\Delta x\\ \Delta y\end{pmatrix}=\frac{1}{f}\begin{pmatrix}\delta x\\ \delta y\mu m\end{pmatrix}z$$

Equation 1 Relation between resolution of the camera and the depth Z.

3.2.9.1 Geometrical consideration for stereo vision for hi-resolution camera (SONY DFW-SX900 1280 x 960)

We will start by processing the picture resolution for the far field navigation with the camera. In our case we have these information:

f : focal length of the lens = 16 mm
δX: Pixel size (=δY) = 4.65 µm
Δx : horizontal resolution.

Z Distance [m]	Δx [m]	Z Distance [m]	Δx [m]
1	0.290625	50	14.531250
2	0.581250	100	29.062500
5	1.453125	200	58.125000
10	2.906250	300	87.187500
20	5.812500	500	145.312500
30	8.718750	1000	290.625000
40	11.625000	3441	1000.040625

Table 4 Calculation of ΔX by the depth Z as shown in Equation 1

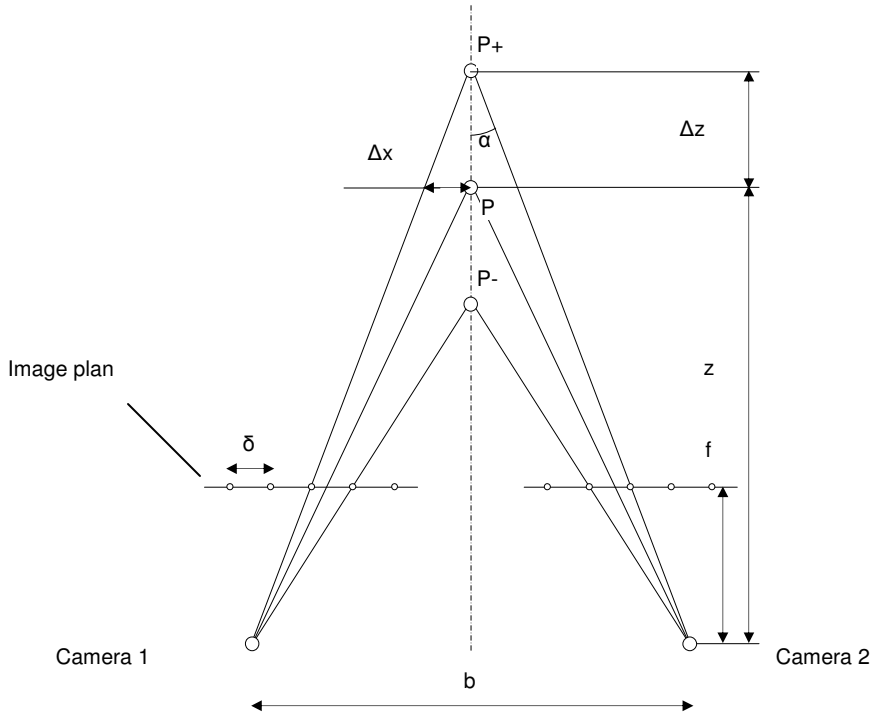


Figure 54 Geometrical representation of the depth resolution Δz

$$\alpha = \tan^{-1}(\frac{b/2}{z+\Delta z}); \Delta x = \frac{z}{f}\delta; \tan\alpha = \frac{\Delta x}{\Delta z}; \Rightarrow \Delta z = \Delta x \frac{z+\Delta z}{b/2} \cong \frac{\delta_x}{bf} z^2$$

Equation 2 Range resolution of a perfect stereo system

Distance [m]	Baseline 0.2 [m]	Baseline 0.6 [m]	Baseline 1.0 [m]	Baseline 1.5 [m]
1	0.0015	0.0005	0.0003	0.0002
2	0.0058	0.0019	0.0012	0.0008
5	0.0363	0.0121	0.0073	0.0048
10	0.1453	0.0484	0.0291	0.0194
20	0.5813	0.1938	0.1163	0.0775
30	1.3078	0.4359	0.2616	0.1744
40	2.3250	0.7750	0.4650	0.3100
50	3.6328	1.2109	0.7266	0.4844
100	14.5310	4.8438	2.9063	1.9375
200	58.1250	19.3750	11.6250	7.7500
300	130.7800	43.5940	26.1560	17.4380
500	363.2800	121.0900	72.6560	48.4380
1000	1453.1000	484.3800	290.6300	193.7500

Table 5 Value calculated with Equation 2

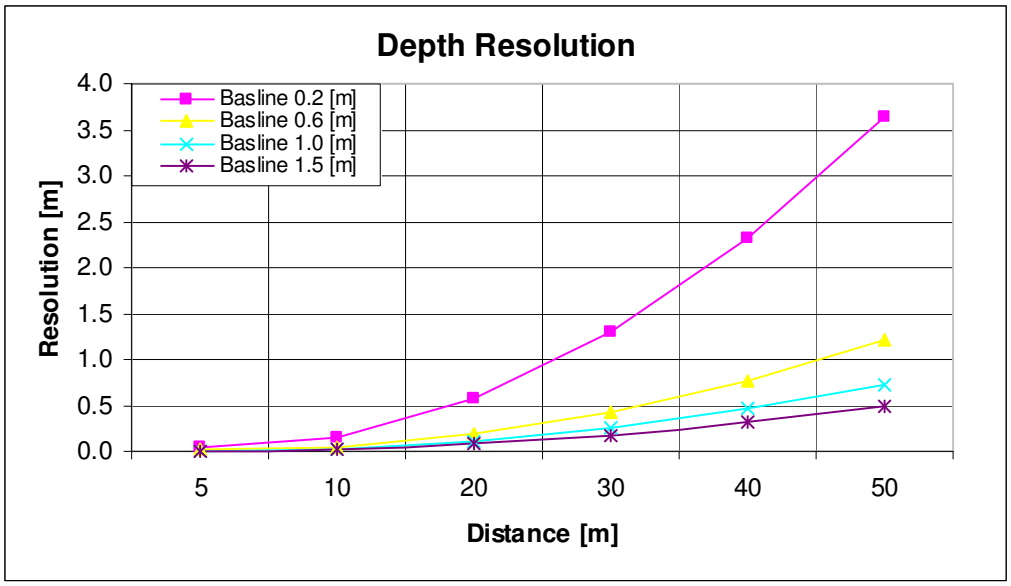


Figure 55 Resolution according to the baseline

The board here shows us the resolutions relative to different types of baseline. If we comment a bit this board we see that at 40.0 [m], the resolution error with a 0.2[m] baseline is of 2.32 [m] although is stays under the 0.5 [m] for a 1.0[m] baseline.

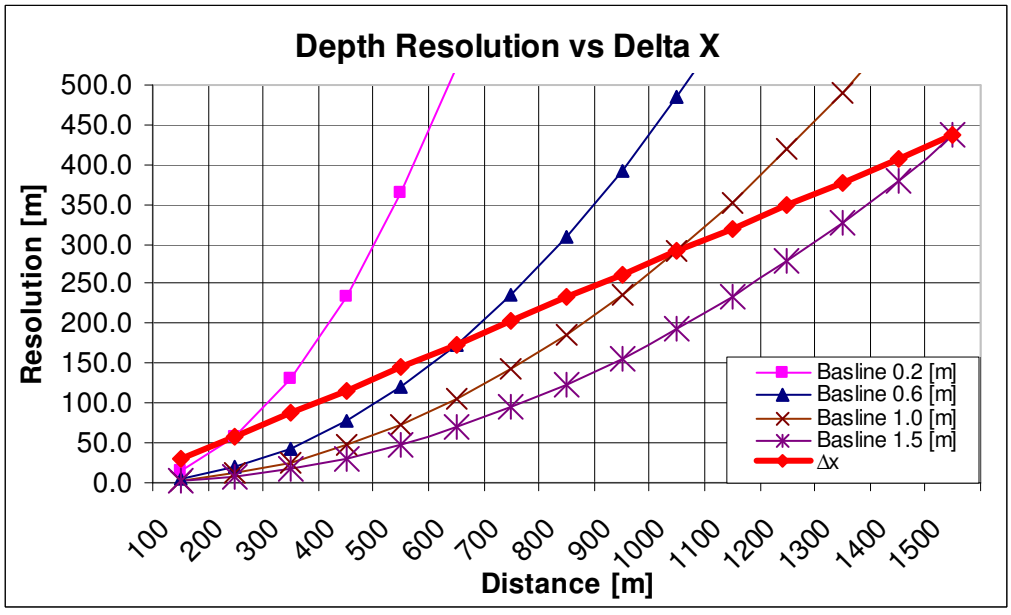


Figure 56 Depth Resolution vs ΔX: cameras resolution according to the stereo resolution. Distances are calculable as long as they are under ΔX, the crossing means maximum value to extract distances. Calibration distances are advice above ΔX.

On Figure 56, let's consider this time the distance at more than one kilometer and a half. The ΔX curve is the maximal resolution calculated by a camera. We can say for example that it is useless to take measures at more than 0.2 [m] (the error being already at about 50 [m]). Between a 0.6 [m] and 1.0[m] baseline, the maximal distance of measure changes from 650[m] to 1050[m]. As our aim was to calibrate the cameras at more than one [km], we chose a 1.0[m] baseline.

3.2.10 Radiometric distortions rectification

Referring to the geometric proprieties of the optics of the cameras, we notice that the distant objects suffer strong distortions the more the projection move off the center of the picture. This is in direct relation with the discretization error of the resolution, which increases with the square of the distance.

To counter this effect, we have decided to apply a paraboloïd filter, which depends on the distance of the object and its deviation from the center (as objects distant from the center look closer than they are). We are satisfied in our case to make a summary correction which can be customizable by changing the paraboloïd factor (in calibration file). A correction within the meaning of [GKA97] [SSBD99] is not useful in our case because we do not use the distant points for navigation, but only has information. Moreover close points (mid field-far field) does not suffer much from the distortion.

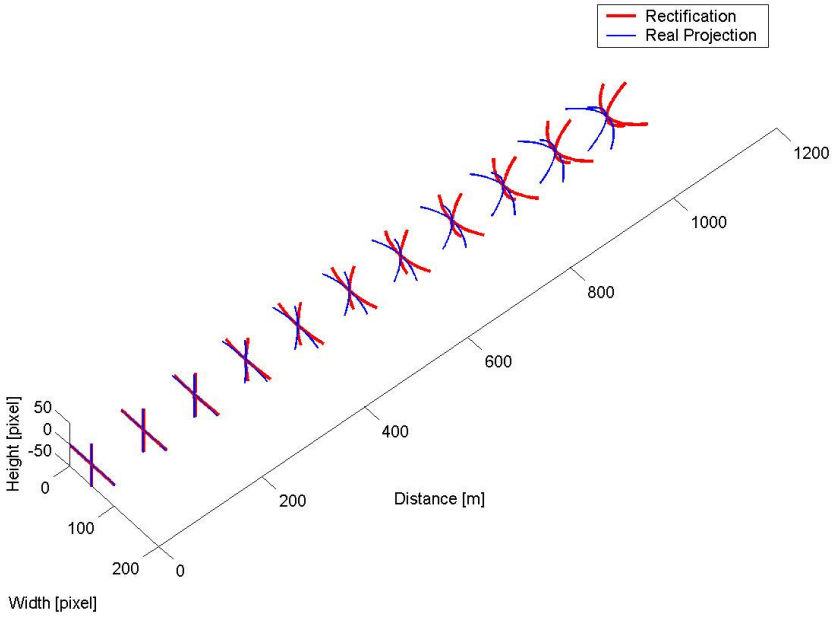


Figure 57 Blue curve the position that appear on the picture, Red curve rectification operated on the blue line

One can see one the picture Figure 58 a picture generated before the radiometric distortion rectification. We can easily distinguish that the point farter than 500 [m] are represented at the wrong depth. The further points are from the center of the picture, the closer they appear.

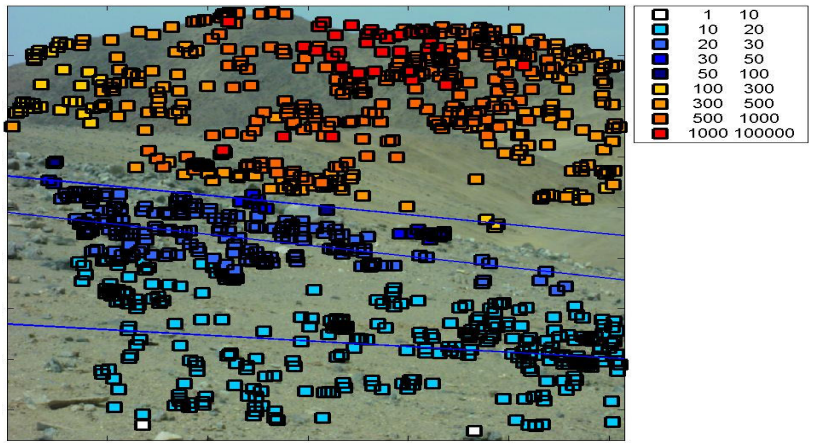


Figure 58 Distances before paraboloidal rectification

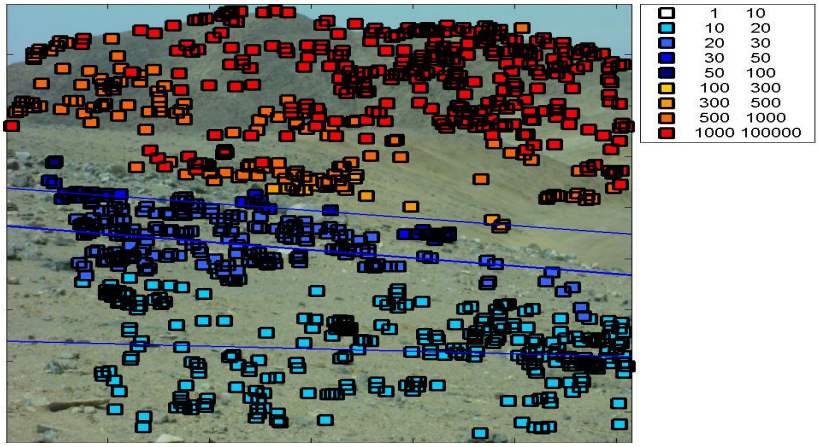


Figure 59 Distances after paraboloïdal rectification

In Figure 59 we se the same picture than above after the rectification. Each point appers to be at the right distance.

It is also particularly interesting to be able to segment the pictures in several zones in order to, for example, analyze the texture in a unique zone; or develop behavior rules, depending on the pieces of visible distances, which could affect the plan of navigation of the rover.

Finally all these information's can, in a very useful way, help the traversability in a case of far field navigation (10-100 [m])

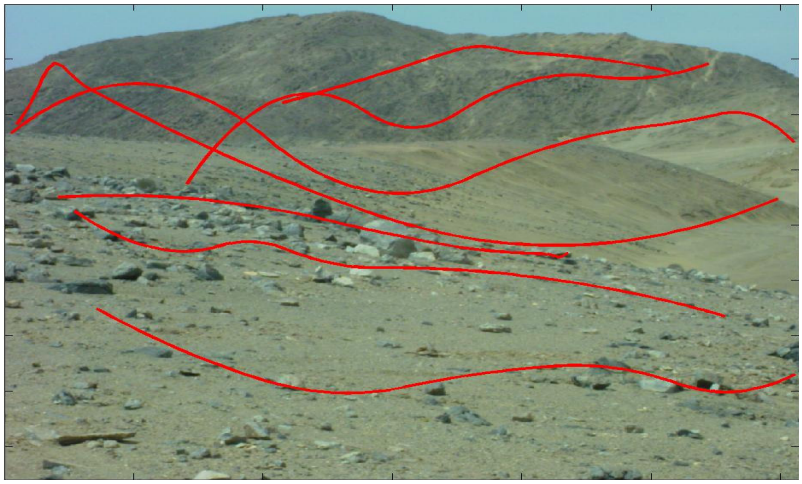


Figure 60 Image segmentation before paraboloidal rectification

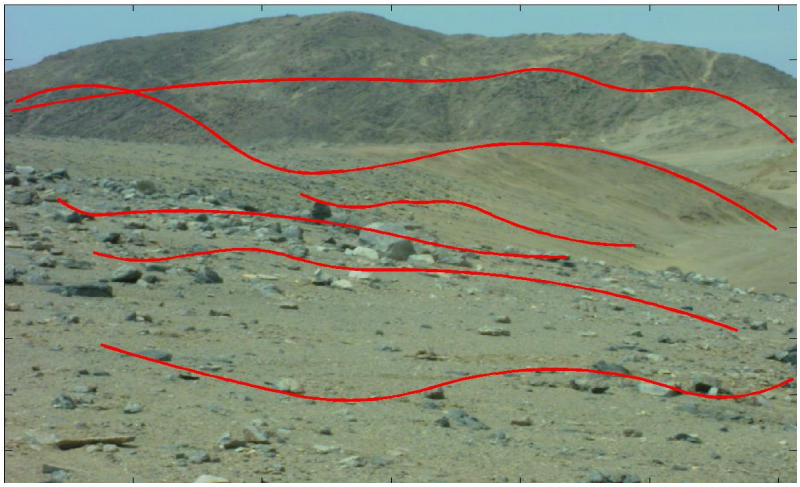


Figure 61 Image segmentation depending by the defined range of distances

On both Figure 60 & Figure 61, the curves have been created by a 5th degree interpolation of points. The curve shows the mid distance of the detected segment on the previous pictures (respectively Figure 58 and Figure 59) And we can see the improvement of the rectification method.

It is certain that others statistical method can be used to have a different segmentation representation. .

3.2.11 3D Modeling

Once we have all the keypoints and their exact positions in space, we can represent them in space for a better understanding of the picture.

The following Figure 62 have been modeling with Matlab by using the coordinate generated by the original picture in Figure 59. It show us the first 0-50[m] keypoints position in the space.

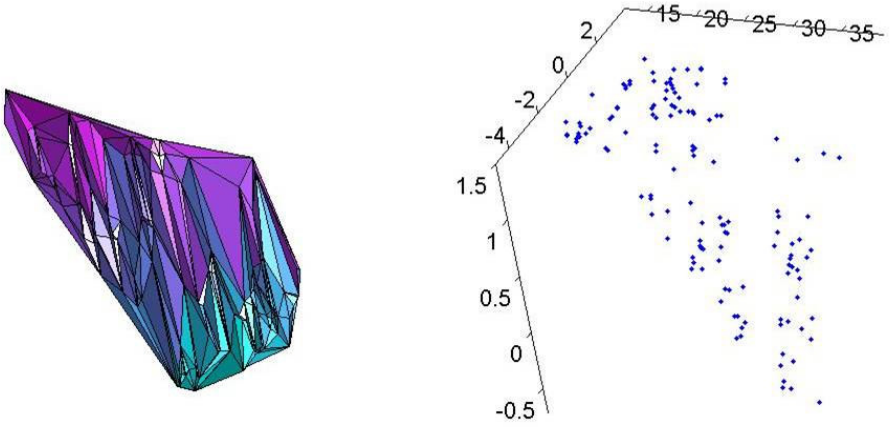


Figure 62 3D representation of the space from 0 to 50 [m] from the cameras, Left: representation with the Delaunay algorithm, Right: Dot representation in 3D space

Better representation of the space can be implemented in the future if needed.

3.2.12 Conclusion

This method can be considered as being a serious alternative to the utilization of a laser on a robot (rover), for distance extraction. Although it has not the same accuracy in the distance calculation, its range is much more relevant. As a matter of fact, the accuracy is in direct relation with the baseline. We can also have a look on the Figure 55: for a 1.0 [m] baseline the error at a 50 [m] distance is lower than at a 0.75[m]distance. Thus, we can consider as very reliable the information's given up to a 250-300[m] distance.

One of the other main advantages is the fact that the system can be self-calibrating without external intervention. This is a very important asset, if one considers that a Hyperion (or Zöé) type rover [LTIA04] can carry out a distance of several kilometers independently, without suffering from violent shocks (which could slightly modify the configurations of the cameras).

In our case, we use a system with identical cameras. SIFT could let us imagine to use stereoscopy with different types of cameras or use a monocular camera, of which baseline is Z (displacement of a robot), without making major modifications to our system.

3.2.13 Result

All result and comments are in Appendix A5 - Stereo vision

3.3 Part III : Data Fusion

We have now an important quantity of information, which describes our initial picture. We need to be able to extract the segments that will be useful and ignore the rest. That is what we call the "Data Fusion". In our case we chose to make two Data fusions: one, which will be oriented "rover safety" and the second, which will be called "science target".

3.3.1.1 Rover safety

We would like to know the parts of the picture, which are dangerous for the rover, in order to be able to advise the rover (TEMPEST : rover's navigation system). This way it could suggest to the rover another possible path to the rover to reach its goal. We are not going to generate a path, but preferably give a graduation of the advised areas and to ones to be avoided.

Input data :

- ***Distances rules map***
- ***Occlusion detector map***
- ***Rock detector map*** & ***Texture classification*** & ***Distance map*** (3.2), for the ***Ground roughness map***

Output data :

- ***Traversability map***

To get a consistent information, in the first place, we are going to sum up information coming from the ***Distances rules map*** and the ***Occlusion detector map*** . This will give us a map with a coefficient of danger for each segment. Then, we will calculate what we will call a ***Ground roughness map*** with the ***Rock detector map*** & ***Texture classification***. We are going to search and then point out the areas where the ground does not seem very flat, and we will balance this result with the ***Distance map***, that is we are going to give less importance to the distant parcels.

Now we can mix the 3 maps together and we obtain areas that are not advised for traversability. Finally we just need to deduce from this result the parts, which are safe. We now that it is more likely to pass as far away possible from a dangerous area.

The idea, which is proposed here is to make an 3rd-degree interpolation in $\mathbb{R}^3$, and use the local extrema for the safe areas and the local minima for the dangerous areas. Thus we will graduate the traversability map from [-100% ; +100%], and the value on the boundary (segments at the boundary), will be at zero (because there is no knowledge of the elements outside the picture).

This way, we have a very clear illustration of the diverse areas.

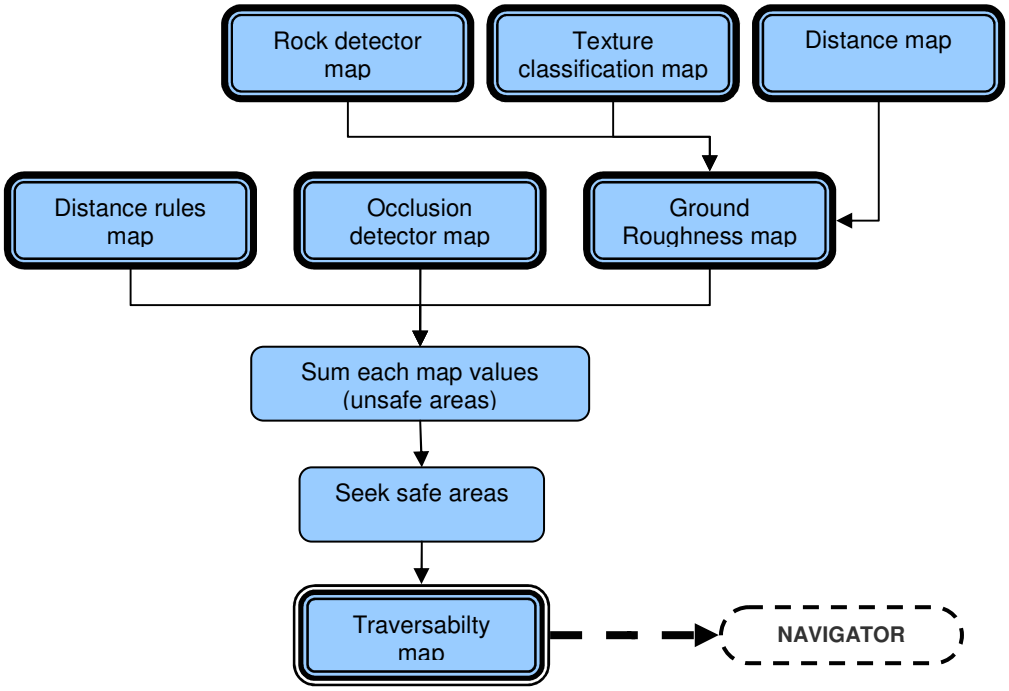


Figure 63 Traversabilty map construction process

3.3.1.2 Science Target

This part will enable us to track areas, which have an important potentiality to be scanned. We are going to generate with the preceding information a map of the interesting areas.

Input data :

- ***Rock detector***
- ***Texture classification***
- ***Color detector map***
- ***Unique color detection map***
- ***Distance rules map***
- ***Fluorescence map*** (not available, biology team)

Output data:

- ***Science interest target map***.

We will simply sum all of the detected value from each maps, with a given balance according to the distance and the science interest.

3.3.1.3 Detectors Summary

Name	processing complexity	Limitations, accuracy
Color detector	*O(n)*	
Unique color detection	*O(s*n)*	
Distance map	*O(s)*	
Distance rules map	*O(s)*	
Ground Roughness map	*O(s)*	Work under 100[m]
Occlusion Detector	aaThreshold *O(2*n) + aaSmooth O(2m²*n) +aaBlob O(n log(n)) => O(n log(n))*	Work under 100[m]
Rock detector	aaThreshold *O(2*n) + aaSmooth O(2m²*n) +aaBlob O(n log(n)) => O(n log(n))*	Work under 100[m]
Fast Fourier Transform	*O(s*i log i)*	
Texture classification map	*O(t*s)*	
Traversability estimation	*O(s)*	Strongly dependant with the segment size.

Table 6 Summary chart of the algorithm's complexity

n = number of pixel in the picture.
i = number of pixel in a segment.
s = number of segment in the picture.
t = number of detected texture
m = size of the smoothing area.

3.3.2 Conclusion

With this manner we not only get clear information about the ground traversability but also important elements relative to the nature of the ground, which surrounds us.

We also have concluded that it was imperative to have a notion of distance to carry out well our study; because in an environment, like a desert environment, it is very difficult, even for the human eye, to distinguish occlusion or even make an estimation of a distance, even being on the site.

The part about "science target" has been partially studied as being out of our scope of study. But it is essential to recall that this part deserves as much investigation and even more than traversability. It has a big exploitation potentiality. Several propositions of picture processing have been carried out but one can easily imagine a lot of other science interests that can be discovered.

3.3.3 Result

All result and comments are in Appendix A2 - Traversabilty estimator results

4 Conclusion

4.1 Contributions

This work has made four major contributions. The first was the application of the SIFT algorithm to stereo vision in support of far field navigation. Second we made use the horizon to assist self calibration. Third we segmented monocular images using the distance information. Fourth we generated an aggregate travaresbilty map using various metrics.

But this work is just the beginning of many other possibilities and gives new ideas for far field navigation. We showed that there is a real interest to know, not only the gross shape of the ground, but also its properties such as texture, roughness and occlusion to make preventive navigation decisions. It is important to emphasize that in this project we covered a wide range of approaches mainly in the toposemantic domain and that future research can easily be added to our system.

4.2 Comments

We used Matlab software to implement all the vision algorithms. This programming language was particularly useful to quickly prototype our algorithms. The execution time was markedly worse than a compiled language such as C/C++ (i.e. semester project) but this was offset by the short development time. I would advise the use of Matlab to at least prototype candidate algorithms to verify the efficiency.

Working on this project was a unique experience. One reason is that "Life in the Atacama" is a large-scale project sponsored by NASA. And the other, it was a different kind of working experience than the ones we can experience at the EPFL. It was based on teamwork with more than 15 people collaborating on the project . Consequently there is a great deal of interaction between team members because all the modules in the system are tied to one another.

4.3 Future Direction

In this project we have, so to speak, laid down the first stone in the construction of a complete far field navigation system. We have experimented with several geometric and toposementic techniques. Each of these detectors individually merits further study, as do the relationships among them.

Before embarking on further research and development, the existing code should be optimized for speed of execution. The main part of the code was written in Matlab, therefore we had to deal with un-complied code (interpreted). A translation phase from Matlab to C++ is imperative because the complete navigating system of the rover is in C/C++.

For future development, it would be very interesting to study the texture detector, which already provides a database of all the types of textures to be detected. One could easily imagine further processing this database for items of scientific interest for example.

The sky detector would also deserve an optimization; we prefer to make a principal component analysis in order to find the best color space in one iteration.

In addition to the traversability map, we could generate an analogous science target map using additional sensors. Measurements of fluorescence, for example, could be processed using additional detectors to generate supplemental information. This extra data would be ignored by the traversability map and only added to the science target map.

Using the concepts that we have outlined, it will be possible to provide intelligent agents a richer view of the physical world. Robots will be able to easily correlate a wide variety of sensor data in a global reference frame. This would allow rovers to be more effective at long distance traverse and science target acquisition.

5 Acknowledgement

I would like to thank particularly Professor Reymond Clavel, who helped me to carry out my diploma project abroad. It was both on the academic and the human level, an unforgettable experience

I also would like to thank deeply Dr. David Wettergreen, who accepted to receive me and carry out this project and who gave me the means to make it in the best conditions.

I would like to thank as well Dr. Charles Baur and Dr. Terry Fong from the "networking team", who, with their efforts and contribution, enabled me to study on a so interesting project.

Finally I would like to thank all the people that helped and supported me.

From life in the Atacama team: Mike Wagner, Domic Jonac, Stuart Heys, Trey Smith and the rest of the team,
Sebastien Grange, Patrick Helmer,
my flatmates Yannick Fournier, Fabien Tache, Steve Burion
my family,
and particularly Emilya Cermak

6 Bibliography

BOOKS Title

[RHOM95] Radu HORAUD et Olivier MONGA , Vision par ordinateur : outils fondamentaux, Published by Editions Hermès 1995.

[DFJP02] David A. Forsyth and Jean Ponce, Computer Vision: A Modern Approach, Published by Prentice Hall 2002

[BSM01] B. S. Manjunath, Color and Texture Descriptors, IEEE Transactions on Circuits and Syxtems for Video Technology, Vol 11, No. 6, 2001, pp. 716-719.

PUBLICATIONS

[CUR03] Chris Urmson, Stéréo vision based Navigation for Sun-Synchronous Exploration, CMU 2003

[KKO992] Kurt Konolige, Stereo geometry, SRI International 1999

http://www.videredesign.com/smallv.pdf

[CGR03] Chauncey Graetzel, Interface utilisateur basée sur les gestes visuels pour chirurgie Analyse du système de vision, Technical Report, EPFL 2003

[KKO991] Kurt Konolige, Small Vision System (SVS), SRI International 1999

http://citeseer.nj.nec.com/konolige97small.html

[AST93] Anthony Stentz, Optimal and Efficient Path Planning for Unknown and Dynamic Environments, CMU 1993

http://citeseer.nj.nec.com/stentz93optimal.html

[AST95] Anthony Stentz, The focussed D* algorithm for real-time replanning, CMU 1995

http://citeseer.nj.nec.com/154664.html

[SSBD99] Sanjiv Singh and Bruce Digney, Autonomous Cross-Country Navigation Using Stereo Vision, CMU 1999

http://www.ri.cmu.edu/pubs/pub_500.html

[AST94] Anthony Stentz, Optimal and Efficient Path Planning for Partially-Known Environments, CMU 1994

http://citeseer.nj.nec.com/stentz94optimal.html

[RSEK94] Reid Simmons and Eric Krotkov, Experience with Rover Navigation for Lunar-Like Terrains, CMU 94

http://citeseer.nj.nec.com/4858.html

[KKKM97] Keisuke Kameyama, Kenzo Mori and Yukio Kosugi, A Neural Network Incorporating Adaptive Gabor Filters for Image Texture Classification, Tokyo Institute of Technology 1997

http://citeseer.nj.nec.com/kameyama97neural.html

[ASMH95] Anthony Stentz and Martial Hebert, A Complete Navigation System for Goal Acquisition in Unknown Environments, CMU 1995

http://citeseer.nj.nec.com/stentz95complete.html

[MBOF93] Michel Buffa and Olivier Faugeras, A stereovision-based navigation system for a mobile robot, INRIA 1993

http://citeseer.nj.nec.com/buffa93stereovisionbased.html

[OCVRM] OpenCV, Reference manual, Intel 2001

http://www.comp.nus.edu.sg/~cs4243/lab/opencv.pdf

[RKO95] R. Koch and M.H. Gross, Visualization of Multidimensional Shape and Texture Features in Laser Range Data using Complex–Valued Gabor Wavelets, ETHZ 1995

http://citeseer.nj.nec.com/gross95visualization.html

[CRA02] Christopher Rasmussen,Combining Laser Range, Color, and Texture Cues for Autonomous Road Following, National Institute of Standards and Technology, Gaithersburg, 2002

http://citeseer.nj.nec.com/rasmussen02combining.html

[CUR031] Chris Urmson, Navigation (Attacama project), CMU 2003

http://www.frc.ri.cmu.edu/atacama/workshop/MA22-03lita.wksp.cu.nav.pdf

[JMSB01] Jitendra Malik, Serge Belongie, Thomas Leung and Jianbo Shi , Contour and Texture Analysis for Image Segmentation, University of California at Berkeley 2001

http://citeseer.nj.nec.com/malik01contour.html

[CFOL00] Clark F. Olson, Larry H. Matthies, Marcel Schoppers, Mark W. Maimone, Robust Stereo Ego-motion for Long Distance Navigation, JPL 2000

http://robotics.jpl.nasa.gov/people/mwm/papers/egomotion.pdf

[DGL99] David G. Lowe, Object Recognition from Local Scale-Invariant Features, University of British Columbia 1999

http://citeseer.nj.nec.com/lowe99object.html

[DGL04] David G. Lowe, Distinctive image features from scaleinvariant keypoints, University of British Columbia 2004

http://www.cs.ubc.ca/~lowe/papers/ijcv03.pdf

[YKR03] Yan Ke Rahul Sukthankar, PCA-SIFT: A More Distinctive Representation for Local Image Descriptors, CMU, 2003

http://www-2.cs.cmu.edu/~yke/pcasift/pca-sift-irp-tr-03-15.pdf

[GKA97] Gerda Kamberova and Ruzena Bajcsy, The Influence of Radiometric Correction on Stereo Matching, University of Pennsylvania 1997

http://citeseer.nj.nec.com/kamberova97influence.html

[CFO03] Clark F. Olson Habib Abi-Rached Ming Ye Jonathan P. Hendrich, Wide-Baseline Stereo Vision for Mars Rovers, Washington 2003

http://faculty.washington.edu/cfolson/papers/pdf/iros03.pdf

[ZZRD94] Zhengyou Zhang, Rachid Deriche, Olivier FAUGERAS, Quang-Tuan LUONG, A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry, INRIA 1994

http://citeseer.nj.nec.com/38312.html

WEBSITE

[LTIA04] Project: Life in the Atacama, FRC-CMU 2004

http://www.frc.ri.cmu.edu/atacama/

[W01] HIPR2, Image Processing Operator Worksheets

http://www.dai.ed.ac.uk/HIPR2/wksheets.htm

[W02] Robert B. Fisher, CVonline: The Evolving, Distributed, Non-Proprietary, On-Line Compendium of Computer Vision

http://www.dai.ed.ac.uk/CVonline/

[W03] Jean-Yves Bouguet, Camera Calibration Toolbox for Matlab, Caltech, 2004

http://www.vision.caltech.edu/bouguetj/calib_doc/

PERSONAL COMMUNICATION

[MDE04] Matthew Deans, Use of the SIFT algorithm for correlating features between heterogeneous camera types, NASA Ames, February 2004.

7 Appendix A Results

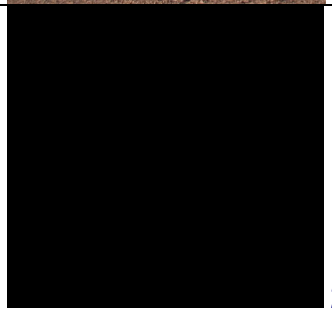
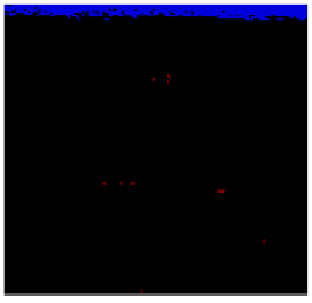
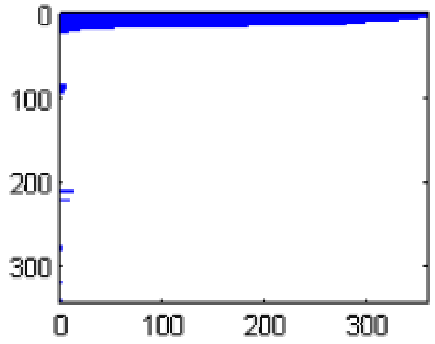
7.1 Appendix A1 - Sky Detector Result

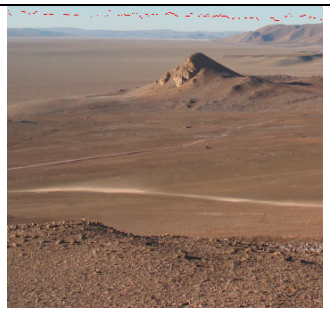
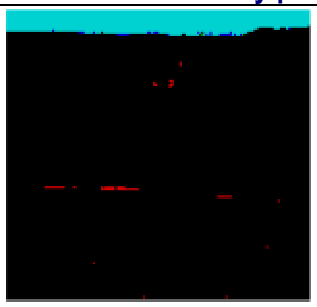
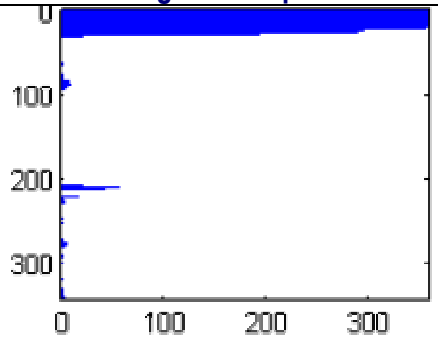
Here, we will show different sky cases, starting from the easier to the more difficult one. All the steps will be showed.

The algorithm aaStep computes several types of information. As we will see in examples below, it reduces the number of colors in the image, but it also computes the presence of the colors in the image, which is represented by a graph in the examples.

It also calculates the presence of each color in order to determine which are the dominant color, and which will be neglected and erased.

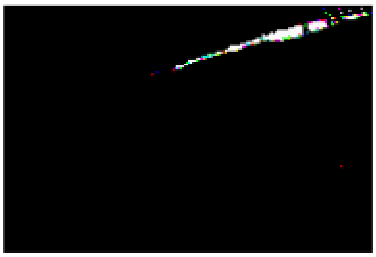
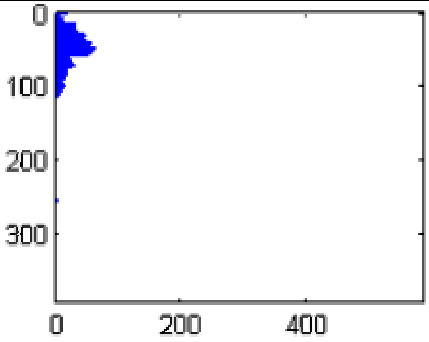
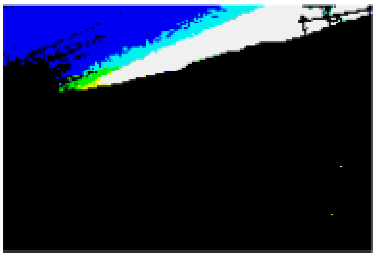
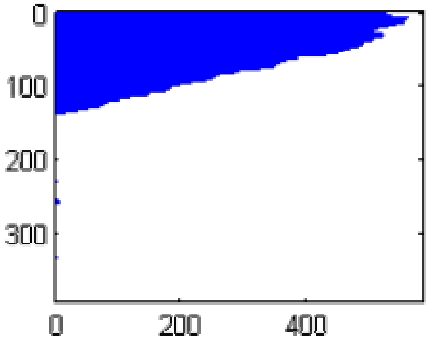
7.1.1 CASE 1 – Standard Case, clear Sky

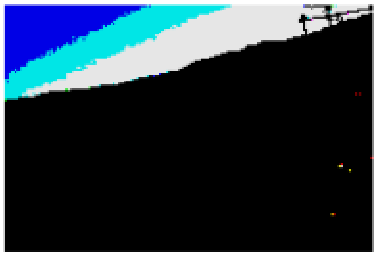
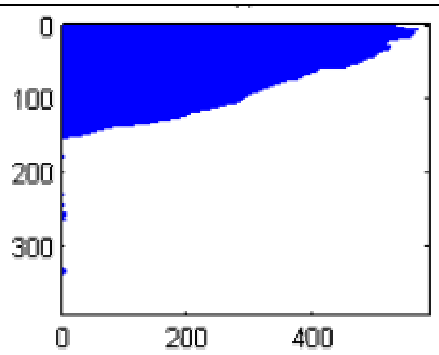
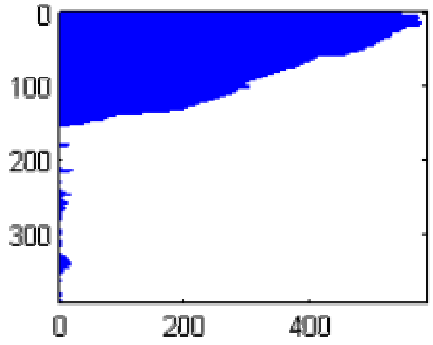
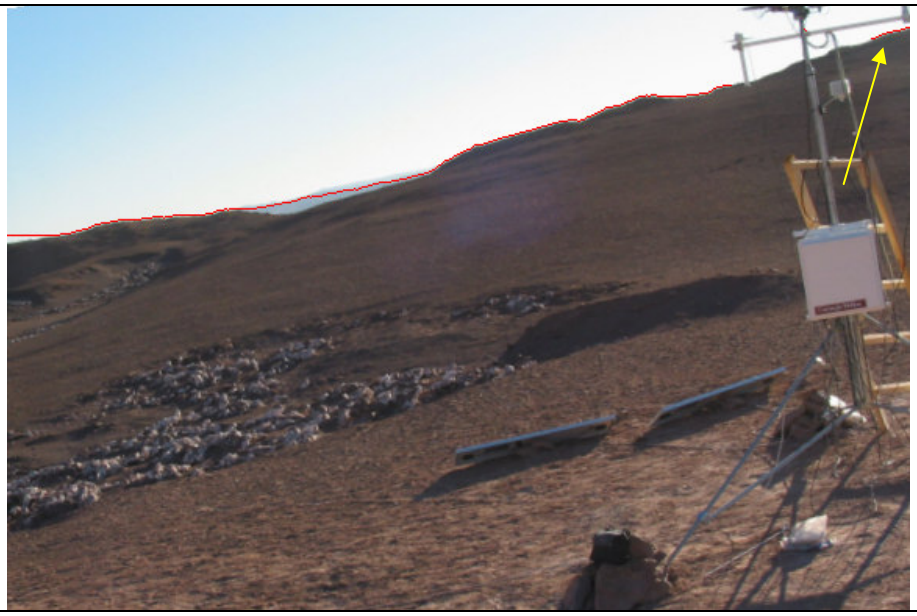
Original	
aaStep algorithm	 250 – 230   220

aaSky algorithm	 **OUTPUT >> Low Sky point information. Change the Step value**
aaStep algorithm	  **210**
aaSky algorithm	

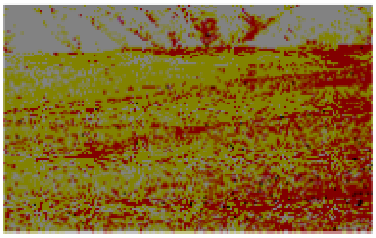
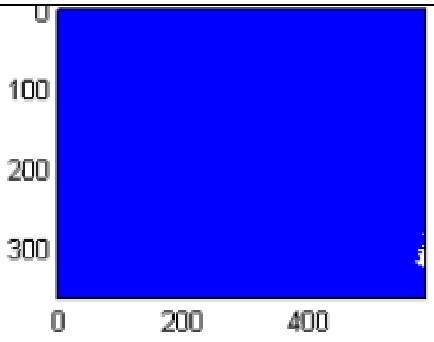
7.1.2 CASE 2 – Medium Case, clear Sky

Now we increase the level of difficulty, we add an important slope to the sky line (rover position) and some obstacles.

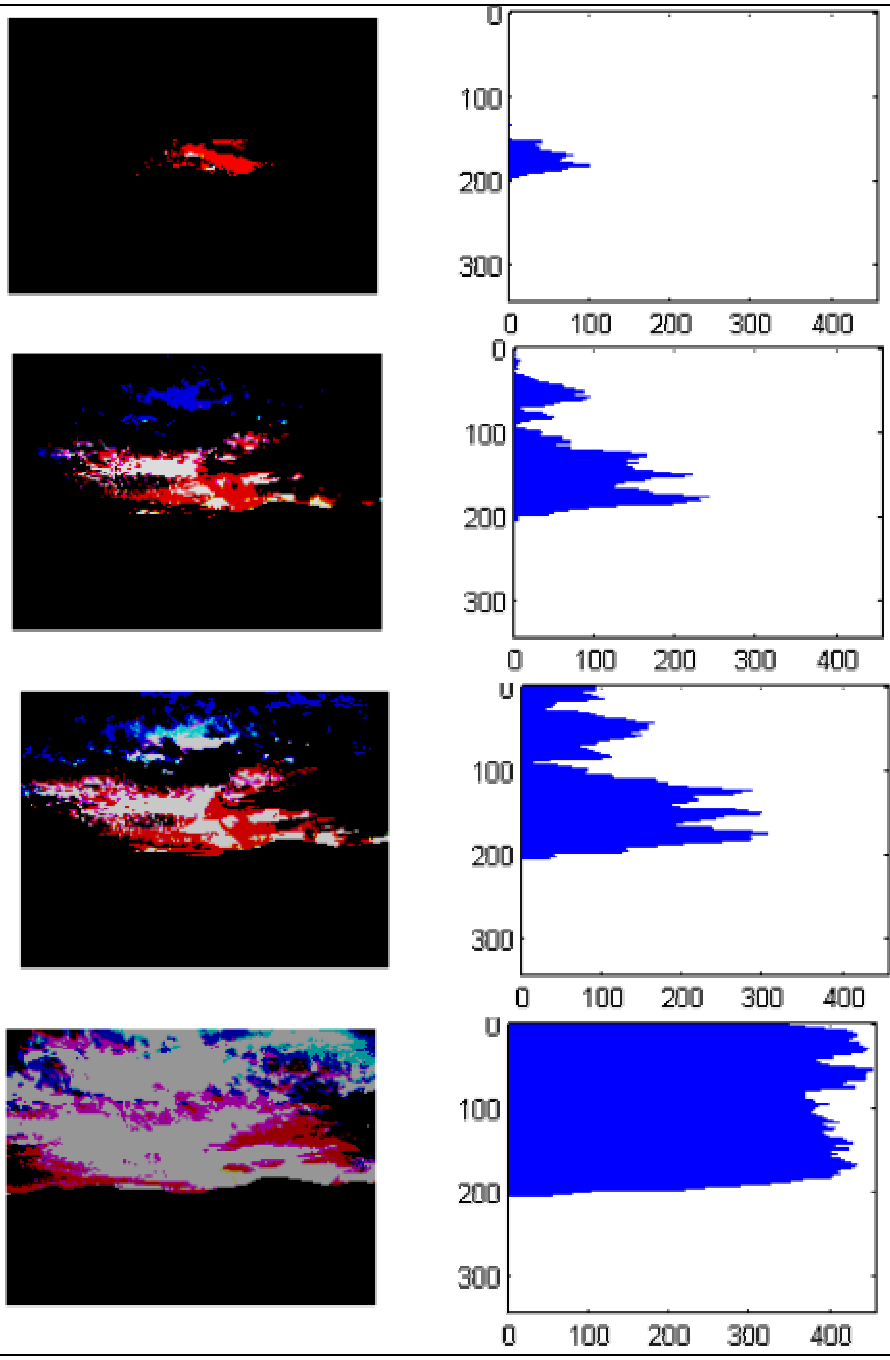
Original	
aaStep algorithm	  **250**   **240**
aaSky algorithm	 **OUTPUT >> Low Sky point information. Change the Step value**

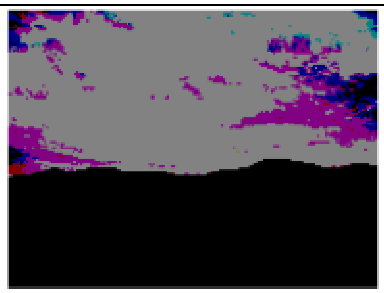
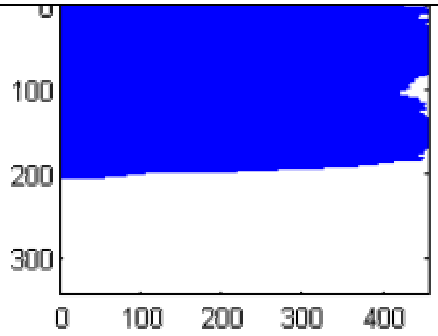
aaStep algorithm	   
aaSky algorithm	

7.1.3 CASE 3 – Medium Case, No Sky

aaStep algorithm	  **250-130**
aaSky algorithm	 **OUTPUT >> No Sky Detected**

7.1.4 CASE 3 – Hard Case, Cloudy Sky

aaStep algorithm	

	 
aaSky algorithm	

7.2 Appendix A2 - Traversabilty estimator results

In order to better illustrate the work which was carried out, we will show, in this section, a complete example of the processing of the traversability map. We begin our analysis with the original images of Figure 64. This image is interesting because it contains various kind of information, like an interesting texture of the ground, a close occlusion lying at distance between 30-50 [m], and moreover it lends itself very well to the calibration.

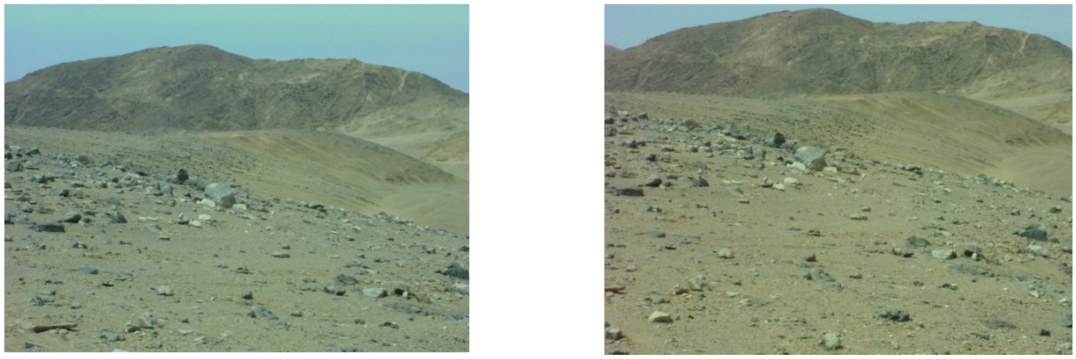


Figure 64 Original Left and Right picture, 0.6 [m] horizontal basline.

We will work only on the left Figure 64 to make all our toposementic image processing's. The original image is a 1280x960 pixels picture, and it will undergo a modification of size which in our case will give us a size of 1229x 860 pixels. The unit of our segments for the toposementic processing is 32x32 pixels, thus we obtain a matrix of dimension 38x26. After having the segmentation, we can fill the Map Distance in the following way:

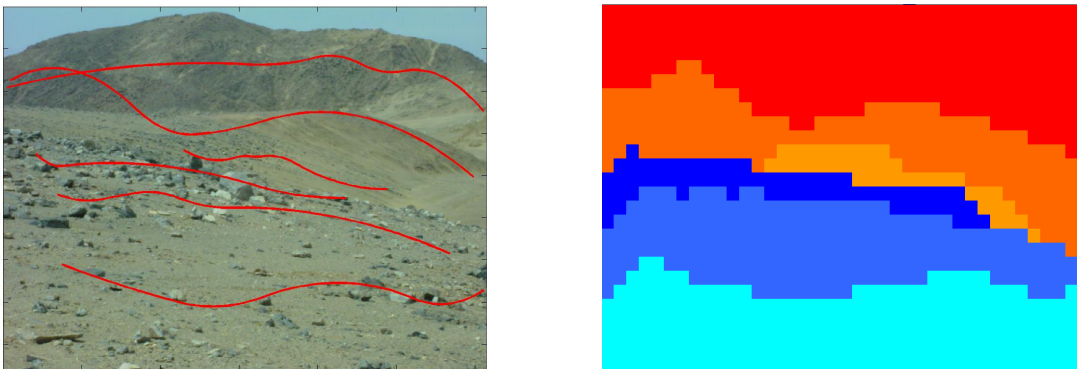


Figure 65 Left : Segmentation of the picture, Right Distance map.

One can notice, by looking at the Figure 65 two of the sections of distance are not present. The Far Field 3 which lies from 50-100[m] and Distant Field 1 (from 100-300[m])

We will apply to the Distance map, the distance rules (seen in 3.1.10), thus to obtain the Distance Rules Map. The result in Figure 66 are voluntarily colored to allow a better understanding. According to Table 2, the segments colored in red have a weight of 100% of danger and the segments in black 75% Danger.

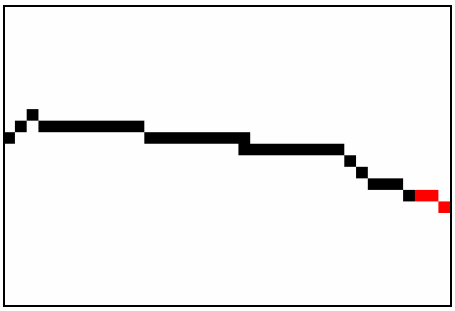


Figure 66 Distance rules map

From this point we use our detector toposementic. It is important to notice that the image will be again resized, but this time only in the image height. We crop the picture at the last far field 3 detected segments. Thus the image makes 38x16 segments now.

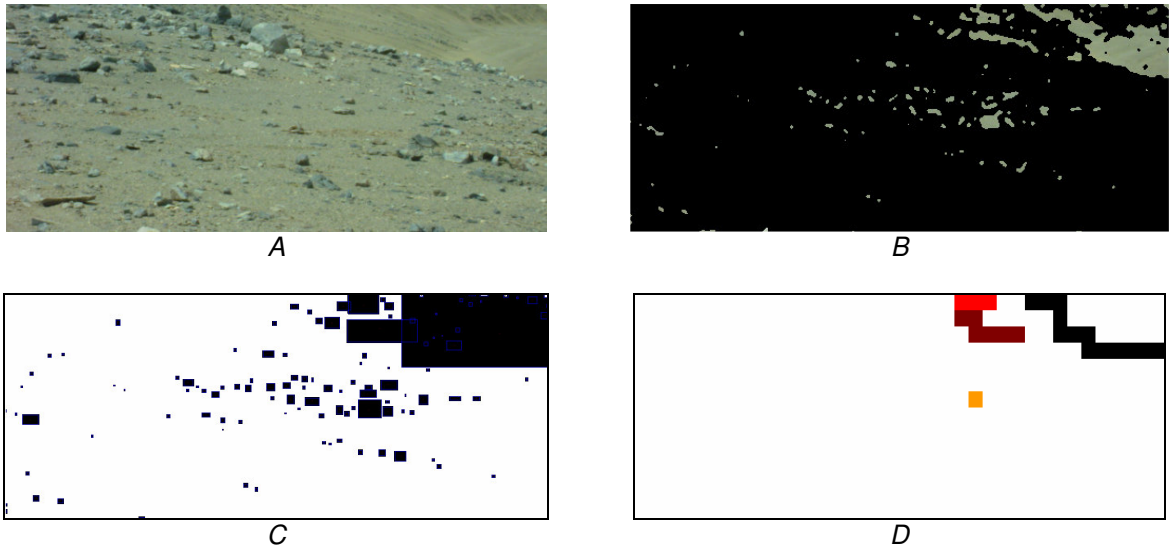


Figure 67 A:Cropped picture from 0 [m] until far field 3: 100 [m], B:Detected occlusion areas, C: Blob detector, surface processing, D: Occlsion Map

Four zone larger than the surface of one segement (> 1024) are detected. Thus we will only consider these zones, all the others are omitted. For better understanding, we have again colored the chart lying in picture Figure 67 D. Surfaces for each of the blob's is as follows: **Blob1** = 1260 pixels, **Blob2** = 1380 pixels, **Blob3** = 2235 pixels, **Blob4** = 18620 pixels. At this point of the processing we have our main two maps.

It Remains to calculate the Ground roughness map, with the Rock detector map & the Texture classification map.

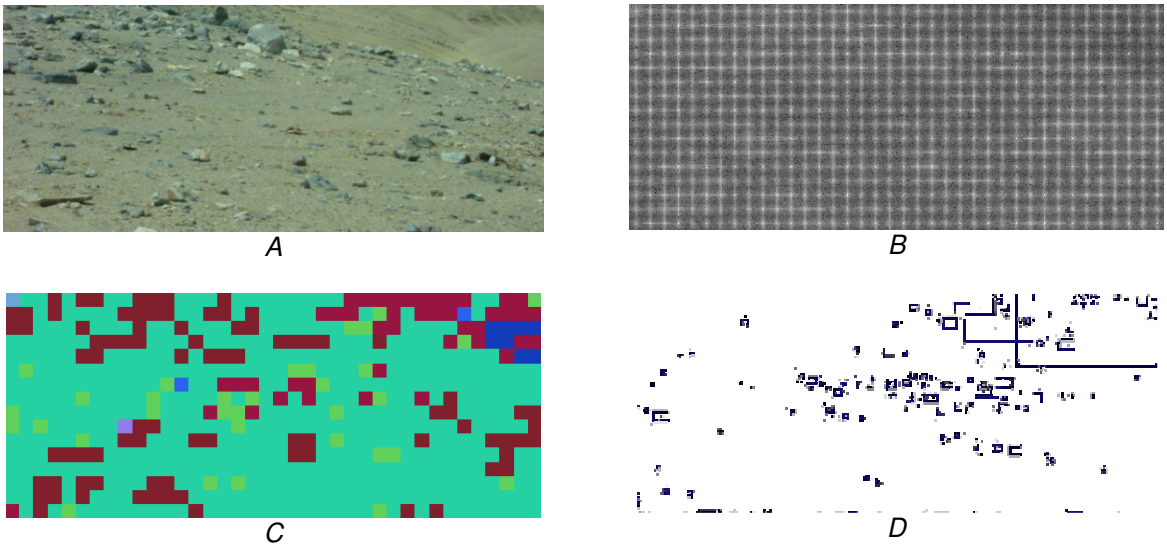
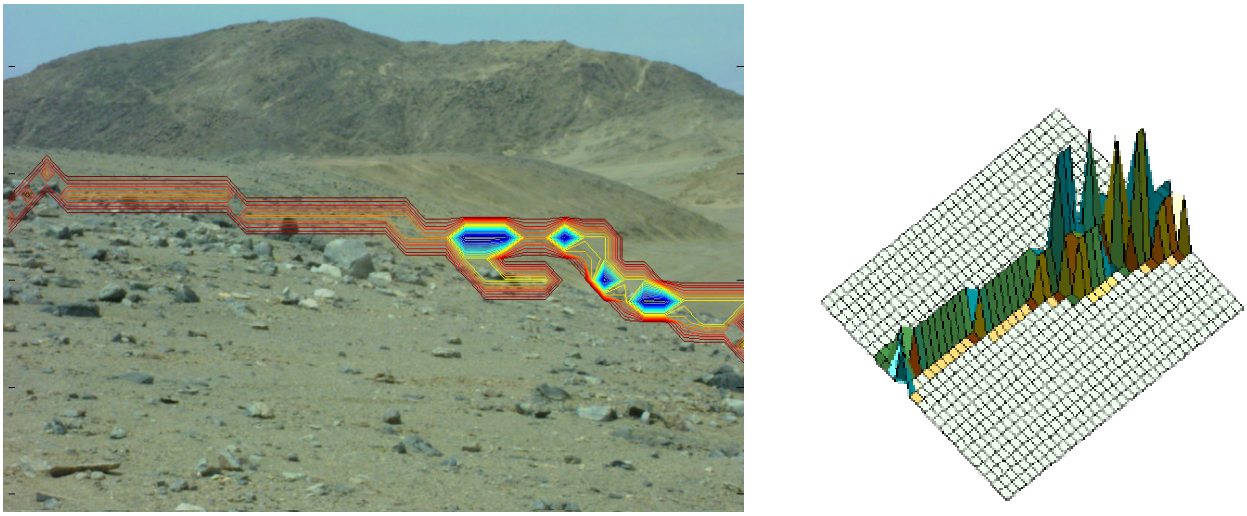


Figure 68 A: Original picture, B: Fast Fourier Transform, C: Detected texture. D: Rock detetcor.

On Figure 68 C the smoothest ground texture are colored in green and light blue. The stony part are in purple. The mix of the 3 map gives us an empty Roughness map, which means that the terrain is quite flat.

At this instant, we have all information to generate the Traversability map. We just have to multiply the value of the segment corresponding has each map and we find the segments with their respective values of traversability.



A B

Figure 69 Illustration of the traversability map, A: each of the visible lines means danger. B: same map but in 3D perspective view, each of the segment correspond to the original picture segment.

Once the Traversabilty map is generated, all the information this have to go to the rover navigation system And he will add this information to the current one to improve the rover autonomous traverse. On Figure 69 (A & B) we have represented the dangerous areas with colored contour. The dark blue contour(Figure 69 A) are the most dangerous part of the picture at the current state of the rover.

7.3 Appendix A3 - Texture classification example

In this example, we have selected 10 pictures (Figure 70) where the field of view goes from 0-100[m]. And we applied on all the pictures the aaFFT2ALL algorithm (see 3.1.8).

Figure 70 Original Hi-resolution pictures

The result of this operation gave us the vector of texture (Figure 71). The idea in this demonstration is to make in a case of future development a filing of all the textures found during a rover traverse or even a whole mission.

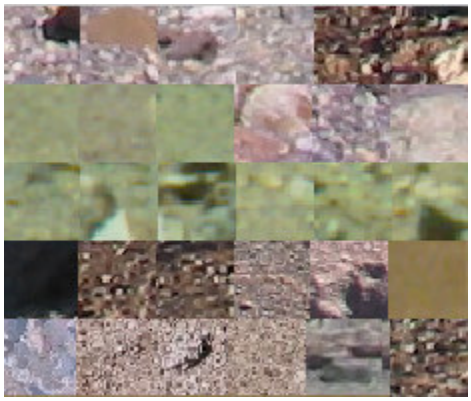


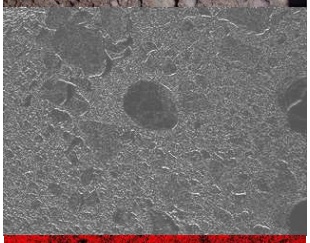
Figure 71 Texture classification result

7.4 Appendix A4 - Rock Detector for rover's underbody

Original

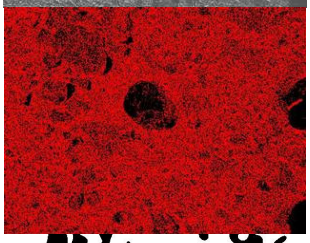
STEP 1 : HIGH-PASS

Take the difference between neighboring pixels. As seen in aaThreshold



STEP 2 : DISCRETIZATION

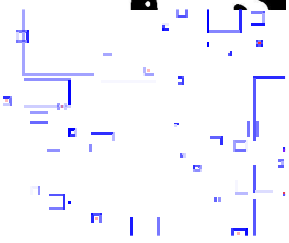
Convert to a binary image by applying a threshold



STEP 3 : DATA SMOOTHING

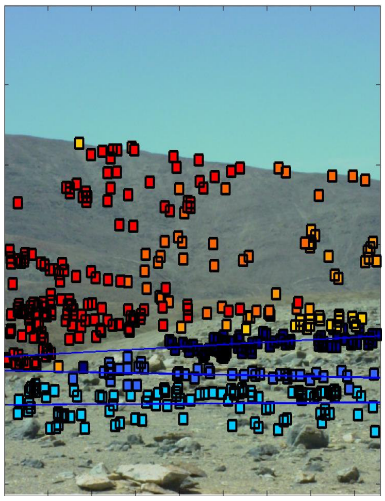
Smooth x3:
Smooth the binary image and fill in small holes.

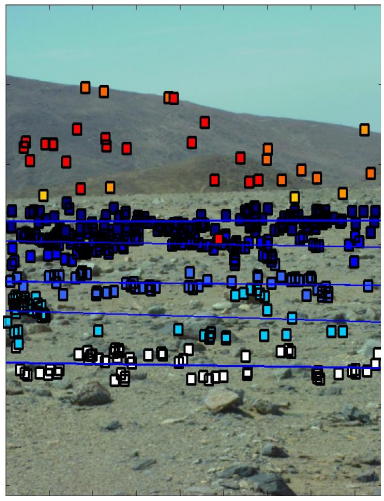
Compute surface and Locate the centroid of each segment.

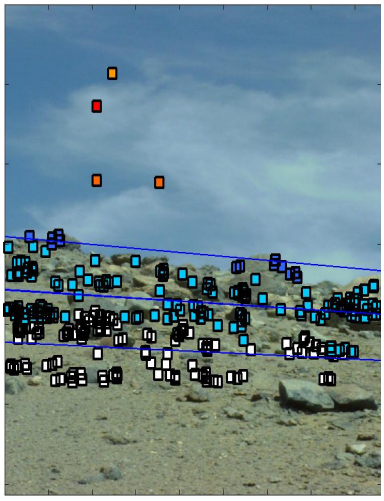


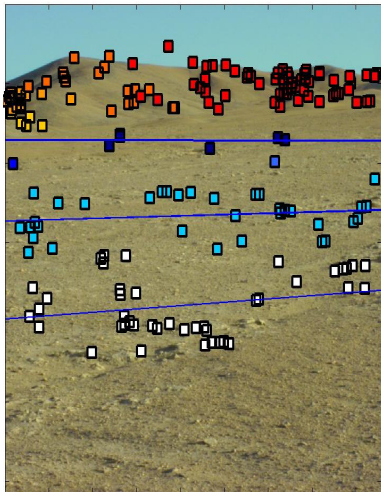
7.5 Appendix A5 - Stereo vision

7.5.1 0.6 [m] Vertical Baseline (Chile)









7.5.2 1.0 [m] Vertical Baseline (Pittsburgh) Distance measurements

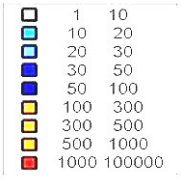


Figure 72 Distance indicator

We have done some distance measurement to se how well the cameras are calibrated. We consider the distances lied on the on the body of the person. The value of the distances are corresponding to the colors that are lying in Figure 72

Real Distance : 14.7 [m]
Measured segment: 10-20 [m]

Real Distance : 19.6 [m]
Measured segment: 20-30 [m]

Real Distance : 24.5 [m]
Measured segment: 20-30 [m]

Real Distance : 24.5 [m]
Measured segment: 20-30 [m]

7.5.3 Unknown baseline (Mars)

We also tested our system on Oportunity's grabbed pictures (Rover on Mars). The value of the distances are not right (Figure 74), because we don't have any information about the cameras intrinsic parameters. Nevertheless we can notice a quite good segmentation map on Figure 73.

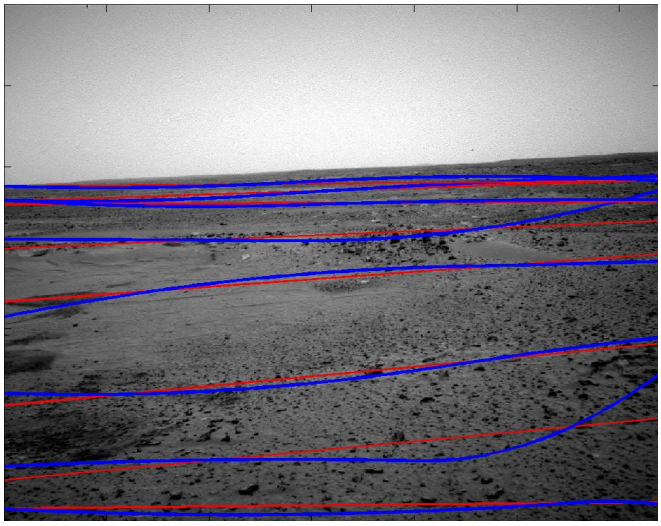


Figure 73 Oportunity's mars landscape picture, Field Segmentation

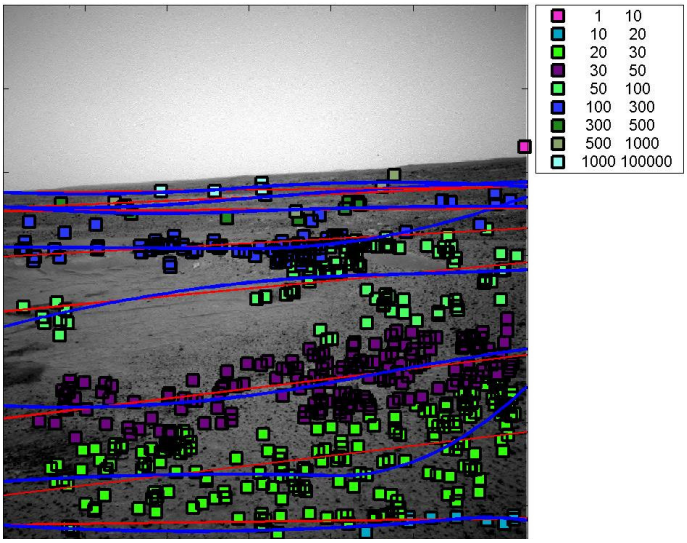


Figure 74 Detected keypoints and distance approximation

8 Appendix B
Main Implemented algorithm

Several algorithm are implemented in this project, but only a part of them are showed here. We also won't mention the algorithm already described in the report.

8.1 Toposemantic

8.1.1 aaFFTMatch
Fast Fourier Transform Matching Algorithm, Seek for best candidate than can be able to enter in the FFT vector database

8.1.2 aaFFT2ALL
Apply Fast Fourier Transform by doing segmentation on whole the picture

8.1.3 aaSkySript
Main script to lauch the Sky Detector

8.1.4 aaScriptNearField
Main script to launch the Rock Detector and the Occlusion Detector

8.1.5 aaSmooth
Smoothing Algorithm

8.1.6 aaScriptTrav
Main function to estimate the Traversability Map

8.1.7 aaThresold
Divide picture from its high frequencies properties, by a given threshold

8.1.8 aaXFill
Seek for blob's on a binary picture and compute it's properties (Size, Centroid, …)

8.1.9 aaXY2
Statistical mass color detector, kernel for the Sky Detector, included in aaStep

8.1.10 Some others Algorithms:
aaBinaries, aaBlur, aaCircleSurface, aaColorDetector, aaEdge, aaHistogram, aaNoiseB, aaNormalize, aaStep, …

8.2 Geometric

8.2.1 aaDepthSegementation

Main Script that make a decomposition of the image by defined segment depending by the depth

8.2.2 aaKeyFusion

Matching algorithm that make a fusion from each of the overlapped keypoints

8.2.3 aaKeyPointDraw

Showing of each of the keypoints

8.2.4 aaKeyStereoCalib

Main script that calibrate the stereo cameras

8.2.5 aaStereoKeys

Main script that recreate the 3D space (in 2D and 3D representation)

8.2.6 aaStereoRectification

Rectification of the radiometric distortion

8.2.7 aaStereoTranslation

Part of the camera calibration, which calculate the translation and the rotation between both pictures.

8.2.8 C: SIFT algorithm

Key.c : Algorithm that seek keypoints

Match.c : Segmentation of the keypoints by a given windows depending if the stereo basline is vertical or horizontal, and Matchig to kind best candidate.

Main.c : Main program for keypoints detector

9 Appendix C Figure Table